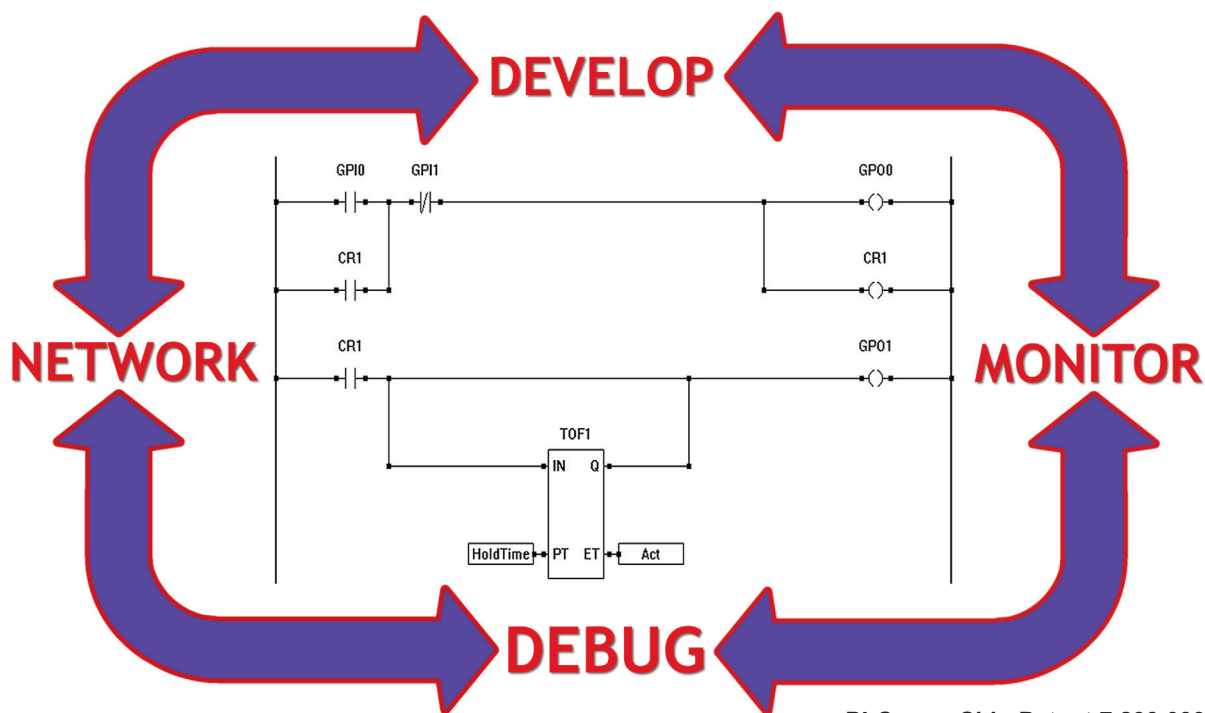


**Standard Edition**

# EZ LADDER

**TOOLKIT**

**ONE TOOLKIT FOR MULTIPLE  
SOLUTIONS**



PLC on a Chip Patent 7,299,099

## USER'S MANUAL

**RELEASE V3.02**

For EZ Ladder V1.0.4.0



# WHAT'S NEW

## RELEASED FEATURES

### V1.0.4.0

#### New Features / Changes / Additions

##### New Features:

1. Added new target : Solves-It! Analog (SI-110, SI-210)
2. Added new target : HEC-2000
3. Now only available COM ports are displayed in the project settings. Now COM ports larger than COM 4 may be used.
4. Changed License Activation from email to web based.

##### Other Changes / Fixes:

1. Corrected minor OptiCAN network items.
2. Corrected issue when placing object J1939\_SPN with Harsh Environment target.

# TABLE OF CONTENTS

This section lists all the sections, and topics included in this manual including page number references.

<b>What's New</b>	TOC-1
<b>Table of Contents</b>	TOC-2
<b>Section 1 - Getting Started</b>	1-1
How to use this Manual	1-2
Installing EZ LADDER	1-2
Activating / Registering EZ Ladder	1-4
Installing EZ LADDER on a Second Computer	1-4
<b>Section 2 - Navigating EZ Ladder</b>	2-1
EZ Ladder Overview	2-2
EZ Ladder Menus	2-3
EZ Ladder Toolbars and Buttons	2-4
Ladder Diagram Workspace	2-5
Cross Reference Palette	2-6
Output Window	2-6
<b>Section 3- Ladder Diagram Basics</b>	3-1
Understanding Relay Logic vs Ladder Logic	3-2
Basic Ladder Diagram Symbols	3-3
Power Rails and Links	3-3
Connection Types	3-4
Understanding How a Ladder Diagram Functions	3-5
<b>Section 4 - Configuring EZ Ladder for Targets</b>	4-1
Project Settings	4-2
Selecting the Hardware Target	4-2
Version Settings	4-3
Options Settings	4-4
Viewing Target Information	4-5
Updating Target Software	4-6
Target Utilities	4-7
<b>Section 5 - Using EZ Ladder to Create Ladder Diagrams</b>	5-1
Creating Ladder Diagram Projects	5-2
Understanding Objects and Variables	5-3
Declaring and Placing Variables	5-3
Variable Types	5-5
Variable Attributes	5-6
Bit Addressable Variables	5-7
Keeping Variable Values on Power Loss	5-8
Placing Objects and Drawing Links	5-9
Copy / Paste	5-10
Inserting / Deleting Rungs	5-11
Saving Ladder Diagram Projects	5-11
Compiling & Verifying Ladder Diagrams	5-11
<b>Section 6 - Downloading and Monitoring Ladder Diagrams</b>	6-1
Switching To/From Monitor Mode	6-2
Connecting to Targets	6-3
Downloading to Targets	6-3
Power Flow Indications	6-3
Scan Time	6-4
Hover Boxes	6-4
Changing Variable Values	6-5

<b>Section 7 - Modbus Slave</b>	7-1
Modbus Slave	7-2
Modbus Configuration	7-2
Register Assignments	7-3
Master Functions	7-3
Communications Errors	7-4
Assigning Registers	7-4
<b>Section 8 - Retentive Variables</b>	8-1
Retentive Variables	8-2
<b>Section 9 - SSI Encoder Input</b>	9-1
SSI (Encoder) Input	9-2
Slave SSI (Encoder) Input	9-3
<b>Section 10 - PWM Outputs</b>	10-1
PWM Outputs	10-2
<b>Section 11 - SPI Slave</b>	11-1
SPI Slave	11-2
SPI Slave Configuration	11-2
Using the SPI Slave Feature	11-3
SPI Timing Diagrams	11-5
<b>Section 12 - Serial Printing</b>	12-1
Serial Print Functionality	12-2
<b>Section 13 - LCD Displays</b>	13-1
LCD Display Functionality	13-2
<b>Section 14 - Matrix Keypad</b>	14-1
Keypad Functionality	14-2
<b>Section 15 - J1939 Communications</b>	15-1
J1939 Communications	15-2
<b>Section 16 - OptiCAN Networking</b>	16-1
What is OptiCAN	16-2
Planning Your Network	16-2
Hardware Requirements & Recommendations	16-2
Twisted Pair Shielded Cable Specifications / Requirements	16-3
Terminating Resistor Specifications / Requirements	16-3
OptiCAN Specifications	16-5
OptiCAN Controller Operation	16-5
OptiCAN Controller Heartbeat	16-5
Configuring a Controller on the OptiCAN Network	16-6
Broadcasting / Sending Data to another Device	16-7
Receiving Data on a Controller from another Device	16-9
The OPTICAN_NODESTATUS Function Block	16-10
The OPTICAN_TXNETMSG Function Block	16-11
Controller Node Registers	16-12
Configuring Other OptiCAN Devices (Non-Controller)	16-13
OptiCAN Network Register Assignments	16-17
OptiCAN Node List Notes	16-18
Sample Network Planning Forms	16-19

<b>Section 17 - Target's Features &amp; Functions</b>	17-1
PLC on a Chip (PLCHIP-M2-1280X) Info	17-2
PLC on a Chip (PLCHIP-M2-2560X ) Info	17-3
PLC on a Chip (PLCHIP-M2-2562X / PLCHIP-M2-2563X) Info	17-4
PLC on a Chip Module (PLCMOD-M2-12800X) Info	17-5
PLC on a Chip Module (PLCMOD-M2-12801X) Info	17-6
PLC on a Chip Module (PLCMOD-M2-25600X) Info	17-7
PLC on a Chip Module (PLCMOD-M2-25601X) Info	17-8
PLC on a Chip Module (PLCMOD-M2-25620X / PLCMOD-M2-25630X) Info	17-9
PLC on a Chip Module (PLCMOD-M2-25621X / PLCMOD-M2-25631X) Info	17-10
Enhanced Baby Bear (ICM-EBB-100) Info	17-11
Enhanced Baby Bear (ICM-EBB-200) Info	17-12
Enhanced Baby Bear (ICM-EBB-300) Info	17-13
Enhanced Baby Bear (ICM-EBB-400) Info	17-14
Enhanced Baby Bear (ICM-EBB-500) Info	17-15
Enhanced Baby Bear (ICM-EBB-600) Info	17-16
Enhanced Baby Bear (ICM-EBB-700) Info	17-17
Harsh Environment Controller (HEC-1000) Info	17-18
Harsh Environment Controller (HEC-2000) Info	17-19
PCS Controllers (PCS-1X0) Info	17-20
PCS Controllers (PCS-1X1 / PCS-1X2) Info	17-21
PCS Controllers (PCS-2X0) Info	17-22
PCS Controllers (PCS-2X1 / PCS-2X2) Info	17-23
Solves-It! Plug in PLC (SI-100) Info	17-24
Solves-It! Plug in PLC (SI-200) Info	17-25
Solves-It! Plug in PLC (SI-110) Info	17-26
Solves-It! Plug in PLC (SI210) Info	17-27
<b>Section 18 - EZ LADDER Reports</b>	18-1
Variable Definitions	18-2
Cross References	18-2
<b>Section 19 - Troubleshooting</b>	19-1
Error Messages	19-2
Connecting Functions to Functions Mistakes	19-5
<b>Section 20 - EZ Ladder Functions &amp; Objects</b>	20-1
Object Basics	20-2
List of EZ Ladder Functions & Objects	20-2
Language Functions & Objects	20-4

# SECTION 1

## GETTING STARTED

---

This section provides detailed information for getting started using EZ LADDER. Included in this section are installation instructions, activating EZ LADDER and instructions on how to use this manual.

## How to Use this Manual

In this manual, the following conventions are used to distinguish elements of text:

<b>BOLD</b>	Denotes labeling, commands, and literal portions of syntax that must appear exactly as shown.
<i>italic</i>	Used for variables and placeholders that represent the type of text to be entered by the user.
<b>SMALL CAPS</b>	Used to show key sequences or actual buttons, such as OK, where the user clicks the OK button.

In addition, the following symbols appear periodically appear in the left margin to call the readers attention to specific details in the text:



Warns the reader of a potential danger or hazard that is associated with certain actions.



Appears when the text contains a tip that is especially helpful.




Indicates that the text contains information to which the reader should pay particularly close attention.

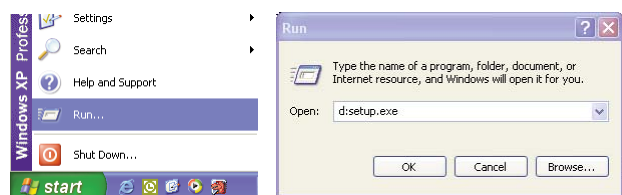
## Installing EZ LADDER

To install EZ LADDER on your computer, follow the following steps. Once EZ LADDER is installed, it must be activated before it may be used with actual hardware targets.

Windows Administrator Rights are required for proper installation. The EZ LADDER directory security is dependent on local / network security settings and may be set for user Read/Execute only. To allow the user to be able to write to this directory, an Administrator must change the permissions accordingly.

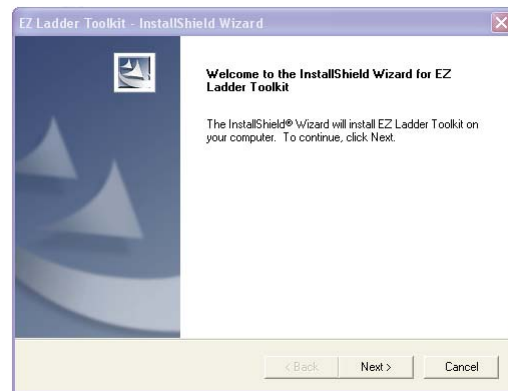
 Windows Administrator Rights are required to install / register / activate EZ LADDER.

1. Insert the EZ LADDER application CD into your CD drive. Select the Start button and choose Run.
2. Select the CD drive that the EZ LADDER CD was inserted. Choose the *setup.exe* and click **OK**.





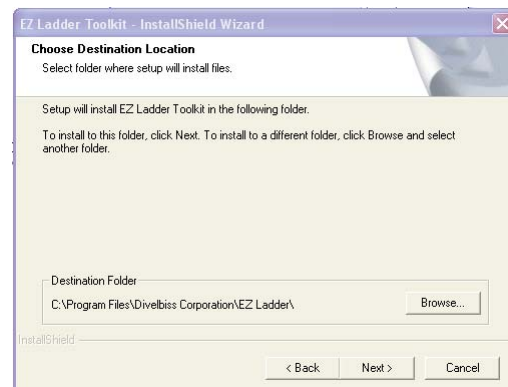
3. This will open the EZ LADDER setup wizard that will install EZ LADDER onto the computer. Click **NEXT**.



4. Complete the Name and Organization fields and enter the Serial Number found on the EZ LADDER CD. Click **NEXT**.

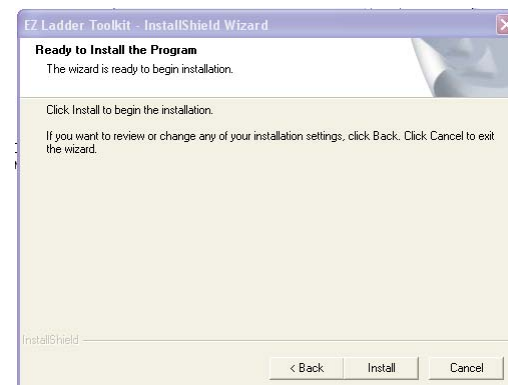


6. Use the default location for installing EZ LADDER or browse and select a different location. Click **NEXT**.



7. All the information is gathered. Click **NEXT** to install EZ LADDER. The EZ LADDER installation will copy all the required files.

Installation will complete. Click **FINISH**.



Before EZ LADDER may be used with hardware targets, it must be registered and activated. Once registered, a license key will be provided to activate the full functionality of EZ LADDER.

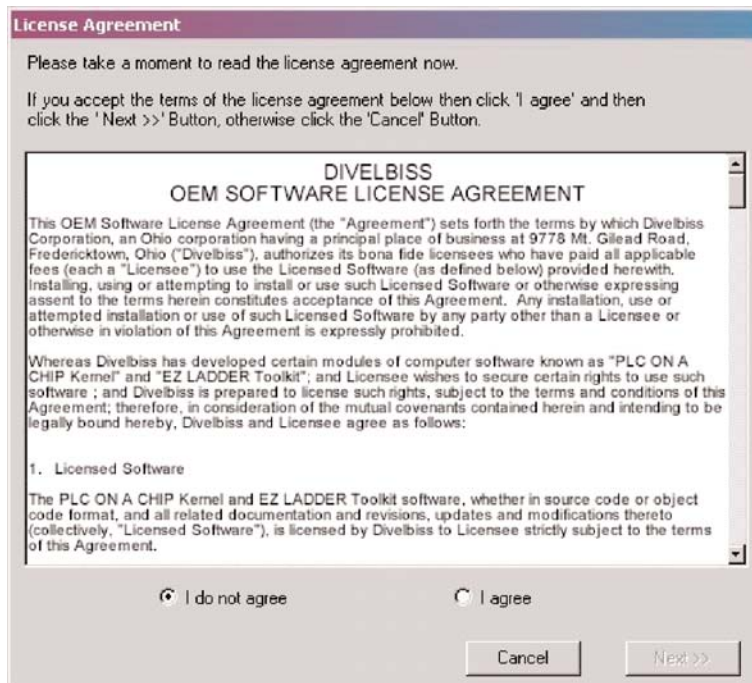
## Activating / Registering EZ LADDER

EZ LADDER will only have demo functionality until it is registered and activated. Once activated, EZ LADDER is fully functional and will operate with hardware targets. The process of registering and activating is completing the on-line registration form receiving a counter key. This key must be loaded into EZ LADDER and will activate the license.

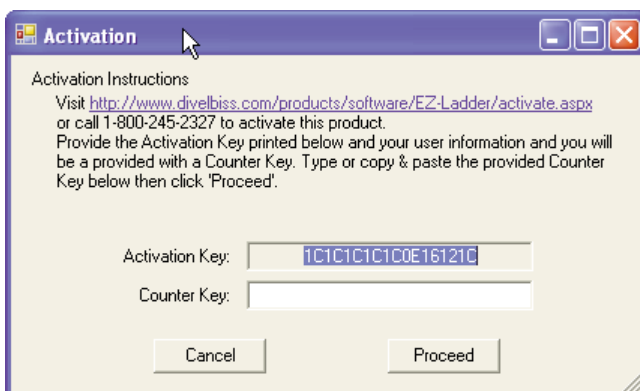
If EZ Ladder is not registered, it will prompt you to do so when the application is started.

To activate / register EZ LADDER, follow the installation wizard as follows:

1. You must read and agree to the license agreement to register the EZ Ladder Application



2. Click the link provided to go to our website. You will need the Activation key provided and your CID Code # provided with your CD. If you do not have your CID Code #, contact Divelbiss Customer Support.



The screenshot shows a Windows-style dialog box titled "Activation". Inside, the text reads: "Activation Instructions: Visit <http://www.divelbiss.com/products/software/EZ-Ladder/activate.aspx> or call 1-800-245-2327 to activate this product. Provide the Activation Key printed below and your user information and you will be provided with a Counter Key. Type or copy & paste the provided Counter Key below then click 'Proceed'."

Below the text, there are two input fields. The first is labeled "Activation Key:" and contains the text "1C1C1C1C1C0E16121C". The second is labeled "Counter Key:" and is empty. At the bottom of the dialog are two buttons: "Cancel" and "Proceed".

3. Copy / Paste or type your Activation key into the Activation key box if not already pre-loaded. Complete all other form entries. All information must be completed.
4. Click the **REGISTER & GET KEY** button. The Activation key and other information will be confirmed and if valid, a Counter key will be displayed. If the information is not valid, the registration will fail.



The screenshot shows the "EZ LADDER Toolkit Activation" web page. The header includes the Divelbiss logo and navigation links: Products, Services, Purchase, Support, Company, and Contact Us. A search bar is also present.

The main content area is titled "EZ LADDER Toolkit Activation" and includes the instruction: "Please complete this form to activate the EZ LADDER Toolkit and receive your activation key."

On the left side, there are several promotional banners: "Looking for a PLC?", "Don't see what you are looking for?", "Now YOU can get all the functionality of a PLC in ONE CHIP", "Enhanced Baby Bear", and "Certified ISO-9001:2000 with Design".

The registration form on the right includes the following fields:

- Name\*
- Email\*
- Company\*
- Phone\*
- Fax\*
- Street\*
- City\*
- State\* (dropdown menu)
- Zipcode\*
- Activation Key: 1C1C1C1C1C0E16121C
- CID Code #

At the bottom of the form is a button labeled "Register & Get Key".

The footer contains links for Home, News, Site Map, Advanced Search, Privacy & Terms of Use Policy, and Terms & Conditions. It also includes copyright information for Divelbiss Corporation 2005 and a contact email for webmaster@divelbiss.com.

5. Copy / Paste the Counter Key that is returned into the 'Activation' dialog box in the Counter Key field. Make sure no extra spaces are present and click **PROCEED**. EZ LADDER will now verify the Counter Key and if valid, will activate the installed copy. Please note, one copy of EZ LADDER may only be registered two times using the web portal (1 desktop / 1 laptop per license agreement). If additional installations are required, please contact Divelbiss Technical Support.

## Installing EZ LADDER on a Second Computer

The EZ LADDER license agreement allows the EZ LADDER Toolkit to be installed on up to two computers (usually a PC and a laptop). To install on a second computer, install EZ LADDER as was done on the first. Follow the same registration steps for the second computer.

# SECTION 2

## NAVIGATING EZ LADDER

This section provides basic information about navigating and working with EZ LADDER tool bars, menus and windows.

## EZ LADDER Overview

The following describes the EZ LADDER application layout including descriptions of each of the sections of the application window.

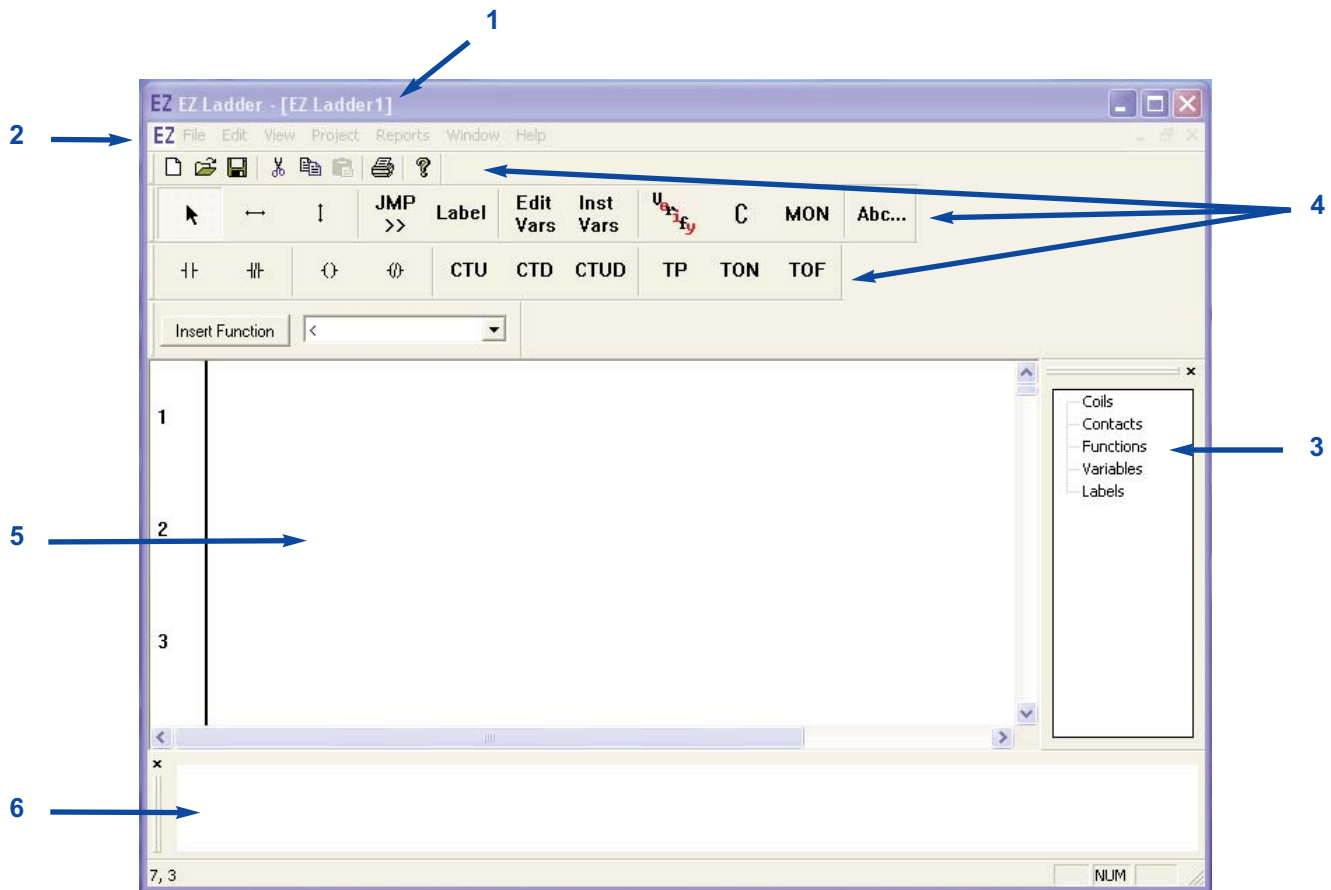




Figure 2.1

- |                              |  |
|------------------------------|--|
| 1. Project Filename :        | The name of the currently open/active EZ LADDER project will be displayed in this position.                    |
| 2. Menus :                   | Drop-down menus for EZ LADDER programming features and options.  |
| 3. Cross Reference Palette : | Clickable Cross References for functions, objects and variables.   |
| 4. Tool Bars :               | Tool bars for placing functions, objects and drop-down function lists.   |
| 5. Ladder Workspace :        | Area where the ladder diagram is drawn.  |
| 6. Output Window :           | This is where status messages are displayed when <i>Verifying</i> or <i>Compiling</i> ladder diagram programs. |

 The Cross Reference Palette and the Output Window may be viewed or hidden. To change the viewing option, Select the **VIEW** menu. Select your viewing options here.

 The Tool Bars may be repositioned by right-clicking on the end of the tool bar and dragging it to a new position. These tool bars may also be closed. To open a closed tool bar, select the tool bar from the **VIEW** menu.

## EZ LADDER Menus

The EZ LADDER has many features and options. Basic commands, features and options are accessed through menus. Figure 3.1 shows the standard EZ LADDER application.

### FILE MENU

<b>NEW</b>	Creates a new ladder diagram project.
<b>OPEN</b>	Opens an existing ladder diagram project.
<b>CLOSE</b>	Closes the currently open ladder diagram project.
<b>SAVE</b>	Saves the currently open ladder diagram project.
<b>SAVE AS</b>	Names and saves the currently open ladder diagram project.
<b>PRINT</b>	Prints the currently open ladder diagram project.
<b>PRINT PREVIEW</b>	Preview the currently open ladder diagram project as it will be printed.
<b>PRINT SETUP</b>	Setup the printer and print options.
<i>Previous Projects</i>	The last few previously opened projects are listed for quick opening.
<b>EXIT</b>	Closes and Exits EZ LADDER

### EDIT MENU

<b>UNDO</b>	Cancels (undoes) the last performed action.
<b>REDO</b>	Restores the last UNDO action.
<b>CUT</b>	Deletes (cuts) the currently selected object.
<b>COPY</b>	Copies the selected object to the Windows Clipboard.
<b>PASTE</b>	Pastes the copied objects from the Windows Clipboard.
<b>SELECT ALL</b>	Actively selects (highlights) everything in the active window.
<b>SETTINGS</b>	Setup parameters for the Display and other features.

### VIEW MENU

<b>TARGET INFORMATION</b>	Opens a window with specific target information including I/O for the selected target.
<b>CROSS REFERENCES</b>	Opens / Closes the Cross Referene Palette.
<b>OUTPUT</b>	Opens / Closes the Output Window.

### PROJECT MENU

<b>SETTINGS</b>	Defines the setting for the current project and/or target along with the available options and features for each.
<b>BOOTLOADER</b>	Provides a tool to update the “kernel” on the hardware target. This only works in Monitor Mode.

### REPORTS MENU

<b>VARIABLE DEFINITIONS</b>	Creates a viewable and printable report with all the variables declared in the active ladder diagram.
<b>CROSS REFERENCE</b>	Creates a viewable and printable report with the selected functions that are in the active ladder diagarm.

WINDOW MENU

CASCADE	Standard Windows features.
TILE	Standard Windows features.
ARRANGE ICONS	Standard Windows features.

HELP MENU

ABOUT EZ LADDER	Displays the current EZ LADDER file versions, and the licensing information.
-----------------	--

EZ LADDER Tool Bars and Buttons

Tool bars are provided for easy and quick access for commonly used objects and actions. There are three basic toolbars. Each toolbar has shortcut buttons for easy access. Below is description of each toolbar shortcut button.

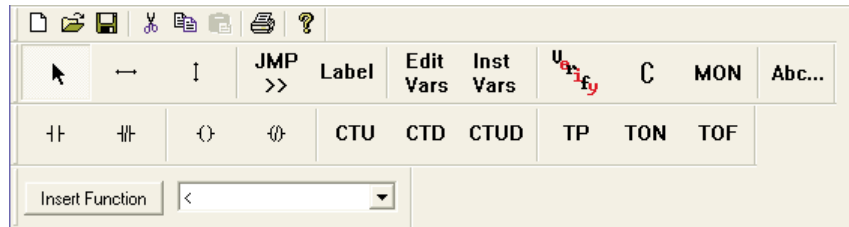


Figure 2.2

<b>NEW.</b> Creates a NEW EZ LADDER Project.	<b>OPEN.</b> Opens the Open File dialog box for selecting EZ LADDER programs.	<b>SAVE.</b> Saves the currently active EZ LADDER Diagram in the workspace.
<b>CUT.</b> Cuts Selected EZ LADDER objects from the workspace.	<b>COPY.</b> Duplicates/Copies the selected object in the workspace	<b>PRINT.</b> Opens the Print dialog box.
<b>ABOUT.</b> Displays information about the EZ LADDER program.	<b>SELECTION TOOL.</b> Selects (Highlights) objects in the workspace.	<b>VERTICAL CONNECTION.</b> Draws vertical connection bars between ladder objects.
<b>HORIZONTAL CONNECTION.</b> Draws horizontal connection bars between ladder objects.	<b>JUMP.</b> Places a JUMP command for the ladder diagram to "jump" to a label. The label must be placed before the JUMP.	<b>LABEL.</b> Creates a Label for the JUMP tool. The label must be created before the JUMP.
<b>EDIT VARIABLES.</b> This opens the edit variables dialog box.	<b>INSERT VARIABLES.</b> Opens the ADD variable dialog box and inserts variables into the workspace.	<b>VERIFY.</b> Verifies the program and elements are complete and do not break any rules.
<b>COMPILE.</b> Compiles the EZ LADDER program to work with the selected Target.	<b>MONITOR.</b> Switches from the workspace to the real-time monitoring screen for program debugging and viewing.	<b>COMMENT.</b> Use to add comments to the ladder diagram project.
<b>DIRECT CONTACT.</b> Places Direct Contact objects in the workspace.	<b>NEGATED CONTACT.</b> Places Negated Contact objects in the workspace.	<b>DIRECT OUTPUT COIL.</b> Places a Direct Output Coil in the workspace.
<b>NEGATED OUTPUT COIL.</b> Places a Negated Output Coil in the workspace.	<b>COUNT UP FUNCTION.</b> Places an Up Counter function in the workspace.	<b>COUNT DOWN FUNCTION.</b> Places a Down Counter function in the workspace.



**CTUD**

**COUNT UP/DOWN FUNCTION.** Places an Up/Down Counter function in the workspace.

**TP**

**PULSE TIMER FUNCTION.** Places a Pulse Timer function in the workspace.

**TON**

**TON-ON DELAY TIMER FUNCTION.** Places a TON function in the workspace.

**TOF**

**TOF-OFF DELAY TIMER FUNCTION.** Places a TOF function in the workspace.



**FUNCTION.** Insert Function Drop down box. Scroll and select the desired function (block).

## Ladder Diagram Workspace

The ladder diagram workspace is the area of the screen where objects and links are placed to create the ladder diagram program. Most objects can be placed at any location in the workspace provided there is actual space available. The *DIRECT* coil, *INVERTED* coil, *LATCH* coil and *UNLATCH* coil are the only objects that must be placed in a particular location. They must be located next to the right power rail. Any attempt to place one of them in another location will cause an error dialog box to be displayed.

A ladder diagram is created using “rungs”. A rung is a horizontal line of logic. EZ LADDER allows the maximum number of rungs to be configured when the target is selected in the **PROJECTS** and **SETTINGS** menu. Figure 2.3 shows the ladder diagram workspace and rungs of horizontal logic.

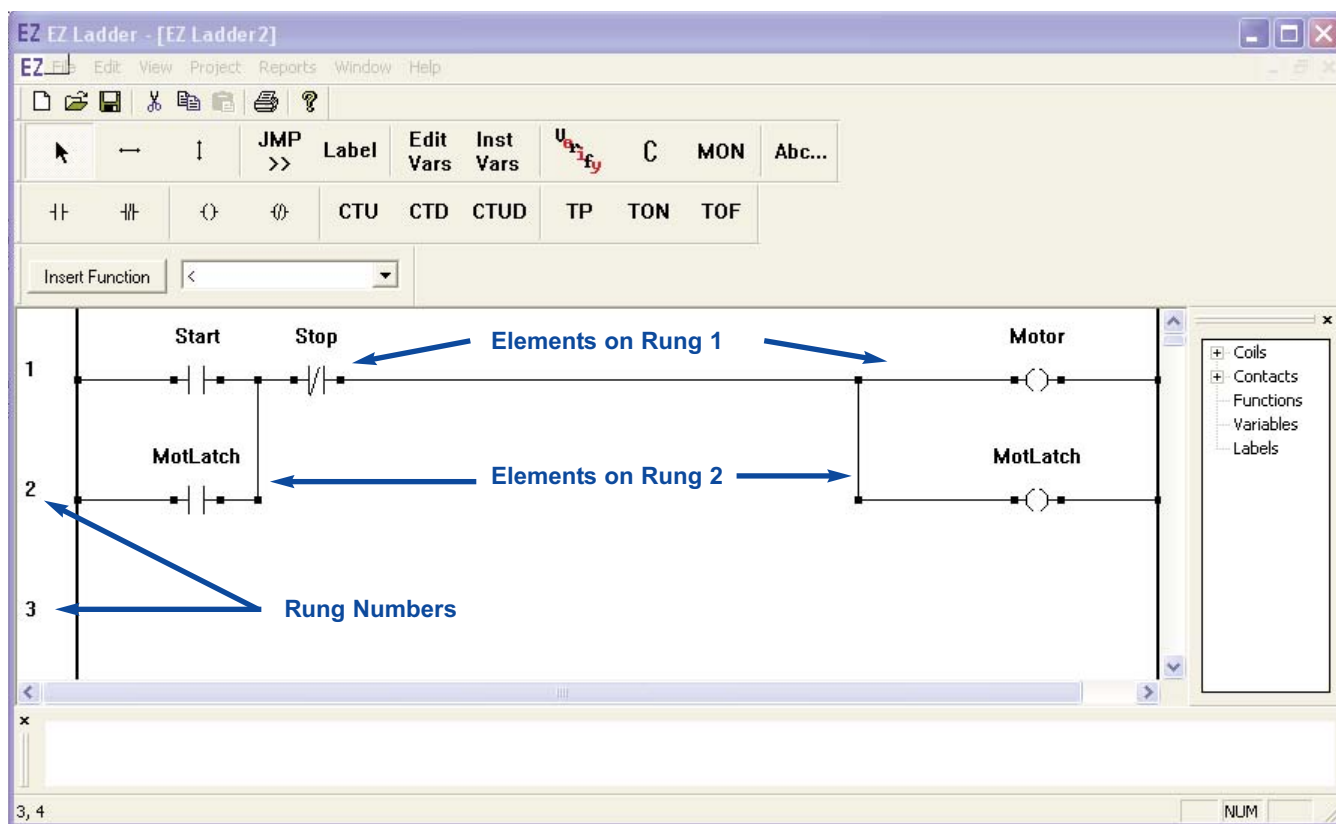



Figure 2.3

## Cross Reference Palette

EZ LADDER provides a real-edit-time Cross Reference Palette. This palette provides lists of contacts, coils, variables, labels and functions as well as their location by rung. This quick reference provides an easy method to locate where a contact or other function is located in the ladder diagram program. Figure 2.4 shows the Cross Reference Window. Cross references are updated automatically when one of the objects changes.

 This palette may be used to find objects quickly. Double-click on any of the object rung numbers, and EZ LADDER will locate and display that section of the ladder diagram.

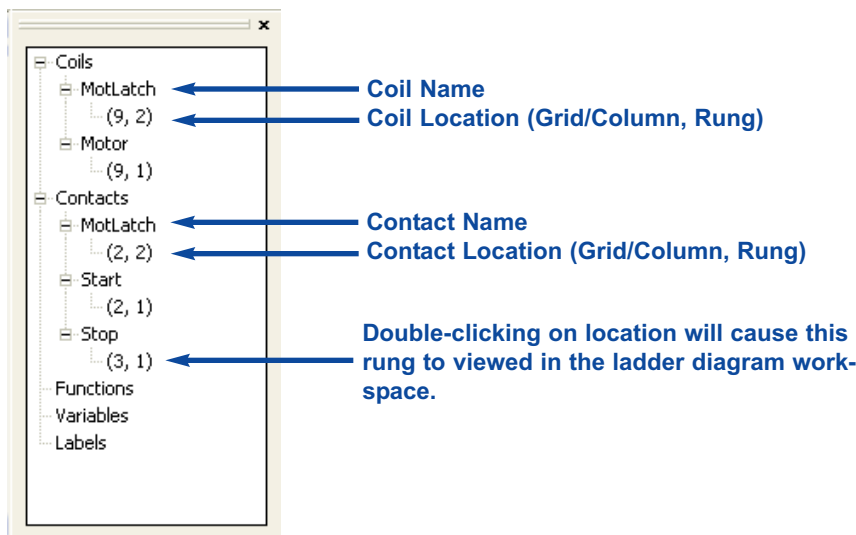


Figure 2.4

## Output Window

The Output window is where ERROR MESSAGES are displayed during the ladder diagram program Verify and Compile operations. Figure 2.5 shows the Output Window displaying error messages.

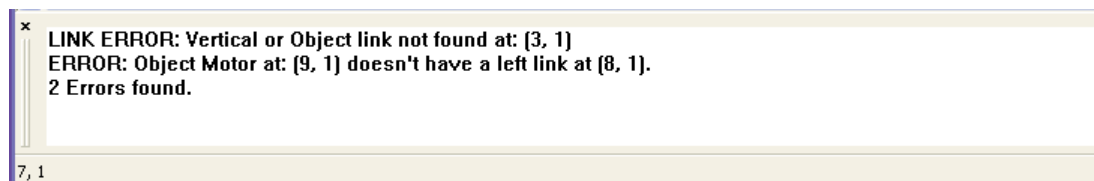


Figure 2.5

# SECTION 3

## LADDER DIAGRAM BASICS



## Understanding Relay Logic vs Ladder Logic

Ladder Diagram (LD) is a graphical representation of boolean equations, using **contacts** (inputs) and **coils** (outputs). The ladder diagram language allows these features to be viewed in a graphical form by placing **graphic symbols** into the program workspace similar to a **Relay Logic** electrical diagram. Both ladder diagram and relay logic diagrams are connected on the left and right sides to **power rails**.

A comparison of a “hard-wired” relay logic system and a “programmable” system using EZ LADDER as the programming platform will show the similarities which make the programming using EZ LADDER quick and easy to apply to any application. Figure 3.1 shows a “hard-wired” relay logic control system. For easy comparison, it is divided into three sections.

**Input Devices:** Includes devices operated manually (ie: pushbuttons) and devices operated automatically (ie: limit switches) by the process or machine being controlled.

**Relay Control Logic:** Consists of relays interconnected to energize or de-energize output devices in response to status of input devices, and in accordance with the logic designed into the control circuit.

**Output Devices:** Consists of motor starters, solenoids, etc. which would control the machine or process.

In place of “hard-wired” relay circuitry, EZ LADDER applications are programmed using “*relay-type symbology*”. This symbology brings ease and familiarity to the programming while adding flexibility.

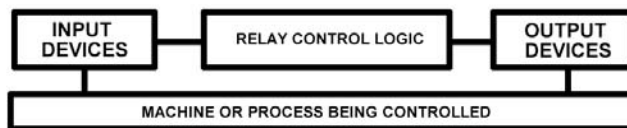


Figure 3.1

The simplicity of the EZ LADDER diagram programming format is shown in Figure 3.2, which illustrates a relay ladder rung as used in relay control systems.

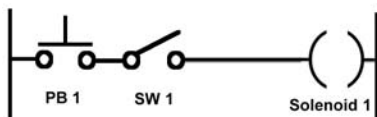


Figure 3.2

The same circuit is shown as its ladder diagram equivalent in Figure 3.3

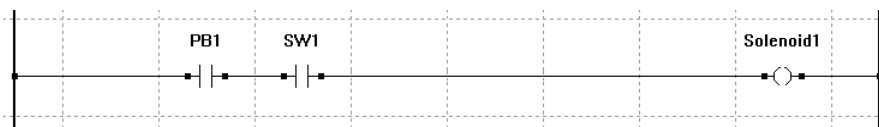


Figure 3.3

## Basic Ladder Diagram Symbols

This section will explain the most basic ladder diagram objects. Section 10 provides definitions and usage for all EZ LADDER Objects.

### DIRECT CONTACT

The direct contact may represent real world inputs or internal relay contacts (CRs). A TRUE condition on the real world input (or CR) will cause the contact to be TRUE. A FALSE condition on the real world input (or CR) will cause the contact to be FALSE.

### INVERTED CONTACT

The inverted contact operates the same as the direct contact, with the exception that the actual logic is opposite of the direct contact. When the real world input (or CR) is FALSE, the contact is TRUE and when the real world input (or CR) is TRUE, then the contact is FALSE.

### DIRECT COIL

The direct coil may represent real world outputs or internal relay coils (CRs). A TRUE condition on the direct coil will cause a TRUE condition on the real world output (or CR), while a FALSE condition on the direct coil will cause a FALSE condition on the real world outputs (or CR).

### INVERTED COIL

The inverted coil operates the same as a direct coil with the exception the logic is the opposite of the direct coil. A FALSE condition on the inverted coil will cause a TRUE condition on the real world output (or CR), while a TRUE condition on the inverted coil will cause a FALSE condition on the real world outputs (or CR).

## Power Rails and Links

### POWER RAILS

A ladder diagram contains rungs with functions and objects. Each rung must be complete for proper operation (should connect to both power rails with continuity using horizontal and vertical links between objects).

Ladder Diagrams are limited on the left and right side by vertical lines, the **left power rail** and the **right power rail**. Figure 3.3 shows a typical diagram utilizing power rails. Figure 3.4 shows the left and right power rail on a ladder diagram rung.

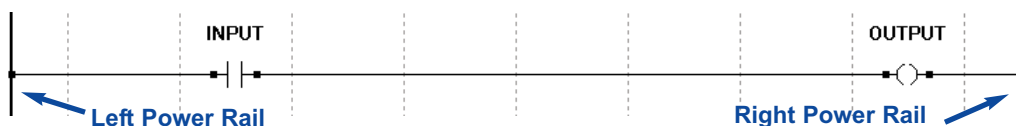


Figure 3.4

### HORIZONTAL / VERTICAL LINKS

Ladder diagram Objects are connected to other symbols or power rails by links. Links may be horizontal or vertical. Each Rung (complete line) contains segments which have a boolean state of TRUE or FALSE based on object status. Any link connected to the left vertical power rail has a TRUE state. Figure 3.5 shows a typical diagram utilizing links.

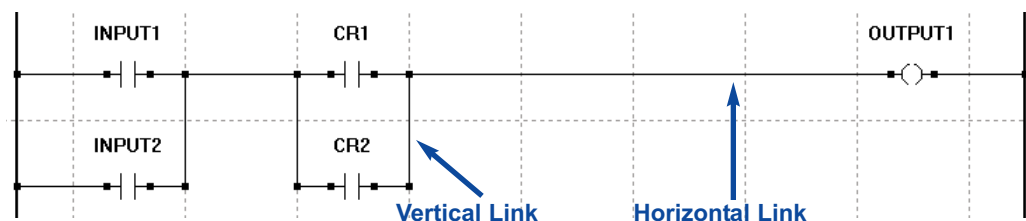


Figure 3.5

### Connection Types

There are multiple types of connections (diagram circuits) that can be created. These are created by combining vertical and horizontal links. Figures 3.6 shows a simple connection. Figure 3.7 shows a series contact circuit. Figure 3.8 shows a parallel contact circuit.

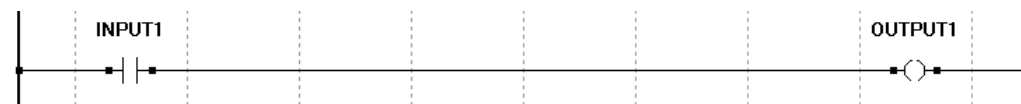


Figure 3.6

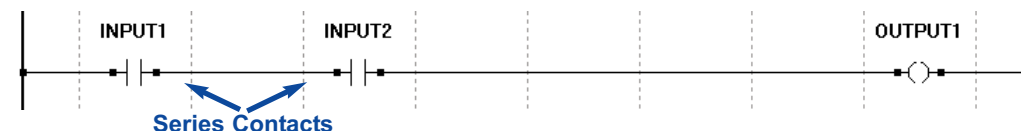


Figure 3.7



Figure 3.8

## Understanding How a Ladder Diagram Functions

When a ladder diagram is installed on a plc or other controller, it will “scan” the program from left to right and from top to bottom. A “scan” is similar to reading a page. A page is read from left to right for each line and then top to bottom (by going to the next line). One complete reading of the program is considered a SCAN. The larger the scan time (one complete read cycle), the less often any I/O is updated. Scan time is an important consideration in the design of a ladder diagram. This scan time may be viewed in the *Monitor Mode* when running a ladder diagram with a hardware target.

Figure 3.9 demonstrates scan of a program.

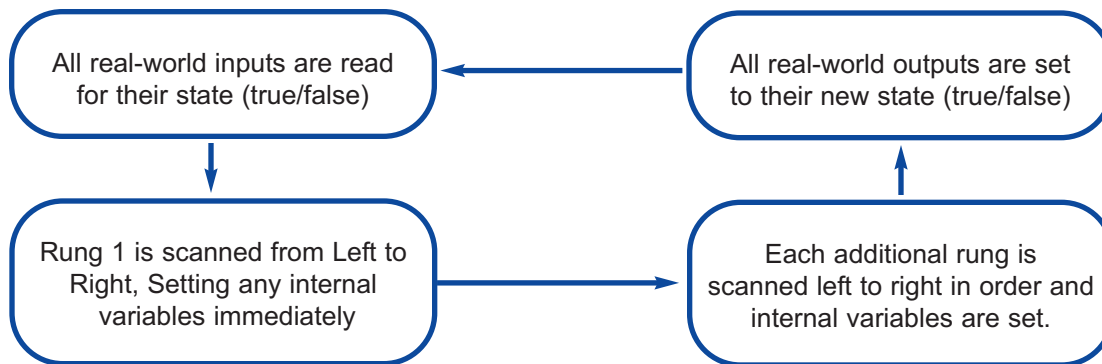


Figure 3.9

# SECTION 4


## CONFIGURING EZ LADDER FOR TARGETS






## Project Settings

EZ Ladder requires the selection and configuration of a hardware target prior to allowing placement of any objects in the ladder diagram. This selection is used to display only objects and functions that are supported by the target. These settings identify the operating parameters and limits of the hardware on which the completed ladder diagram will operate. This setting should match the actual hardware that the ladder diagram will be installed on.

 The *target* may be selected using the **PROJECT....SETTINGS** menu or the *Project Settings* window will be automatically displayed when an object placement is attempted (if a target has not been selected).

 PLEASE NOTE, actual dialog boxes and options will vary depending upon which actual target is selected.

The *Project Settings* window is divided into three individual tabs, TARGET, VERSION, OPTIONS. See Figure 4.1.

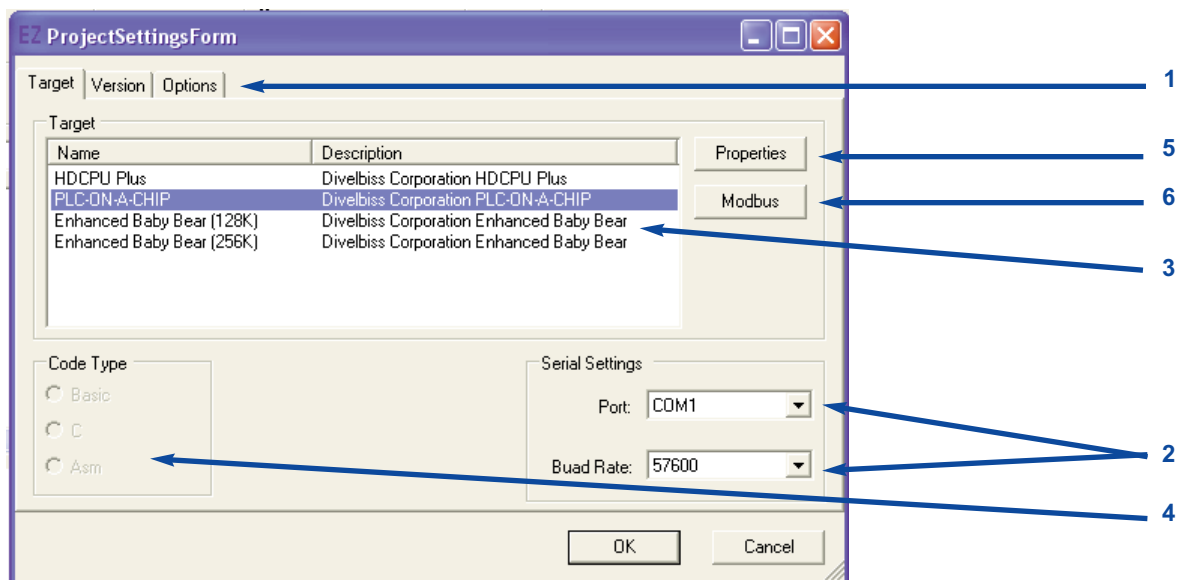


Figure 4.1

1. Project Setting Tabs: Select the appropriate tab to configure target settings. Figure 4.2 represents the *TARGET* tab, Figure 4.3 represents the *VERSION* tab, and Figure 4.4 represents the *OPTIONS* tab.
2. Serial Settings: Configure the Serial Port to what will be used to communicate to the target. These settings are used to connect, download and monitor ladder diagram programs running in EZ LADDER's program run and monitor mode.
3. Target: Select the hardware target for the EZ LADDER diagram project.
4. Code Type: Configure the type of code for EZ LADDER to generate. Refer to specific target requirements or contact the hardware manufacturer. Setting this option is not available on all hardware targets.
5. Properties: Display/Configure specific properties for each specific target selected. This button is only available on certain targets. Properties may include: J1939, PWM and OptiCAN.
6. Modbus: Display/Configure Modbus settings for hardware target. This button is only available on certain targets.

Selecting the Hardware Target

Select the hardware target from the list. Certain targets (PLC on a Chip) require additional configurations. Once the target has been selected (highlighted), if any additional configuration options are available, additional buttons will be visible (**PROPERTIES**, **MODBUS**, etc.). Please refer to the hardware targets datasheet or manual for detailed information regarding configuring the target. If additional configuration is required, it is recommended to complete it at this time.

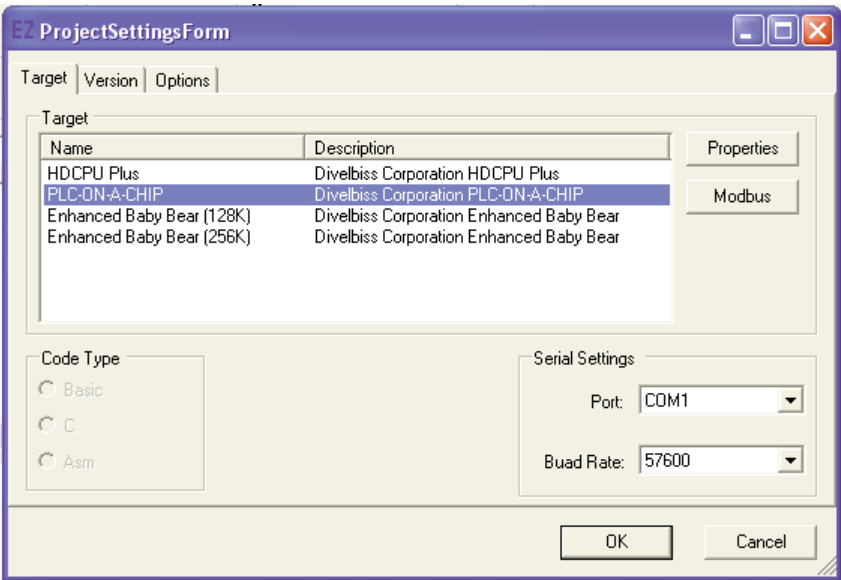


Figure 4.2

Version Settings

The Version settings dialog box will display the current build and version of the ladder diagram that is currently active in EZ LADDER. The build and version information is useful when determining if a program version is current. Figure 4.3 shows the Version settings dialog box. Version setting may be changed in this window, if required.

1. Version Number:
- A version number for the ladder diagram may be entered here. This number will not change, unless edited.
2. Build Number:
- The current build number is displayed here. Each time the ladder diagram program is *Compiled*, the build number automatically increments. This number may be over-written in this window if required.

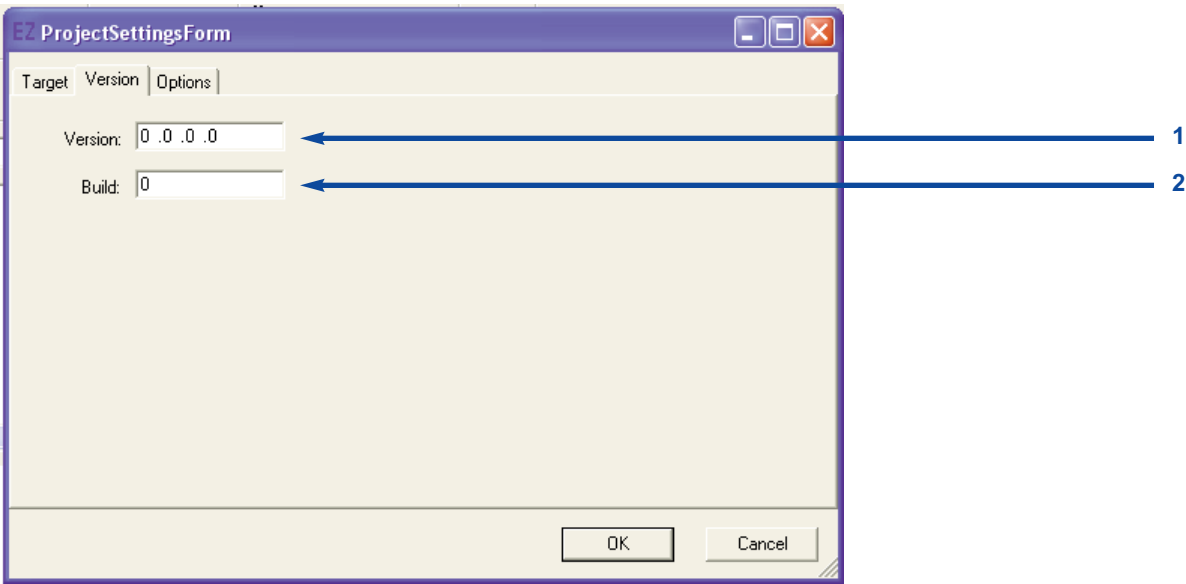


Figure 4.3

Options Settings

The Options settings dialog box will display the currently selected options and preferences. Some of these options are target specific while others are standard. Figure 4.4 shows the Options settings dialog box.

1. Number of Rungs:
- This is where the maximum number of rungs in the ladder diagram is specified. The default number is 100 rungs.

Many of the options and settings are target hardware specific. Only available targets and options will be displayed.

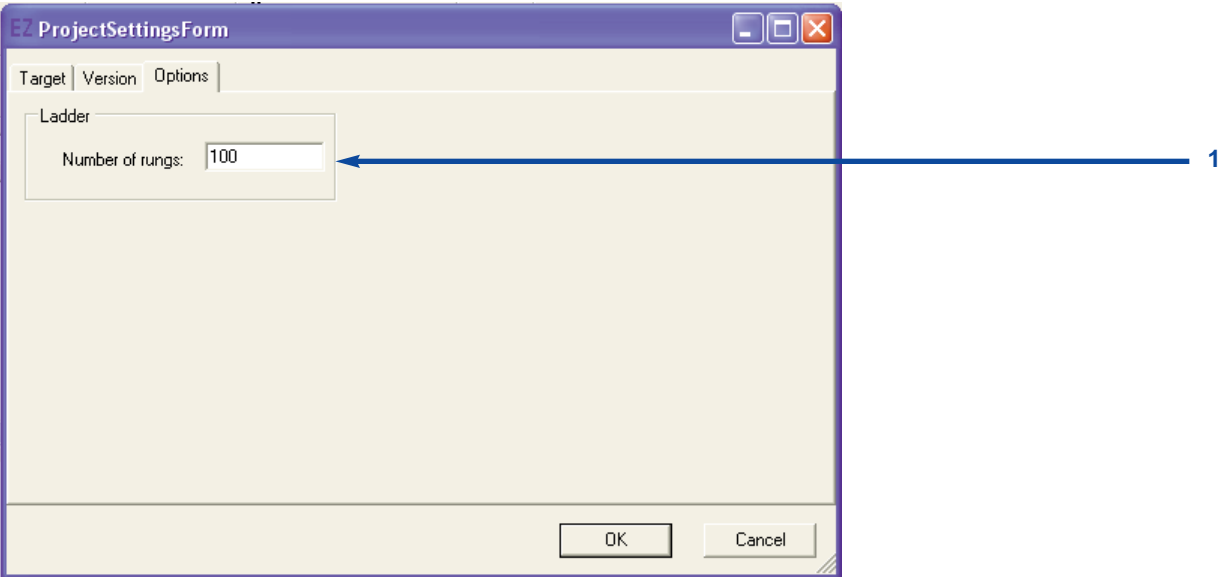


Figure 4.4

The typical “plc” configuraton scans I/O at the beginning / end of a scan as does EZ LADDER. This typically requires structuring of the ladder diagram in a specific order to achieve correct functionality and interaction with different functions and sections of a ladder diagram.



This functionality also provides a true method to control when coils and contacts will be set. For example, say your requirements include that the program must complete one scan before any I/O changes may be made, placing a Control Relay at the end and using it as a trigger to allow everything else to function can accomplish this. During the first scan, the CR contacts are not true; when at the end of the ladder diagram scan, the CR coil becomes true causing the CR contacts to be true on the next scan.

## Viewing Target Information

Specific target information can be viewed using the menu by selecting **VIEW** then **TARGET INFORMATION**. This window will show I/O assignments and other useful information for accessing and programming specific hardware target features. Figure 4.5 shows the *Target Information* window.

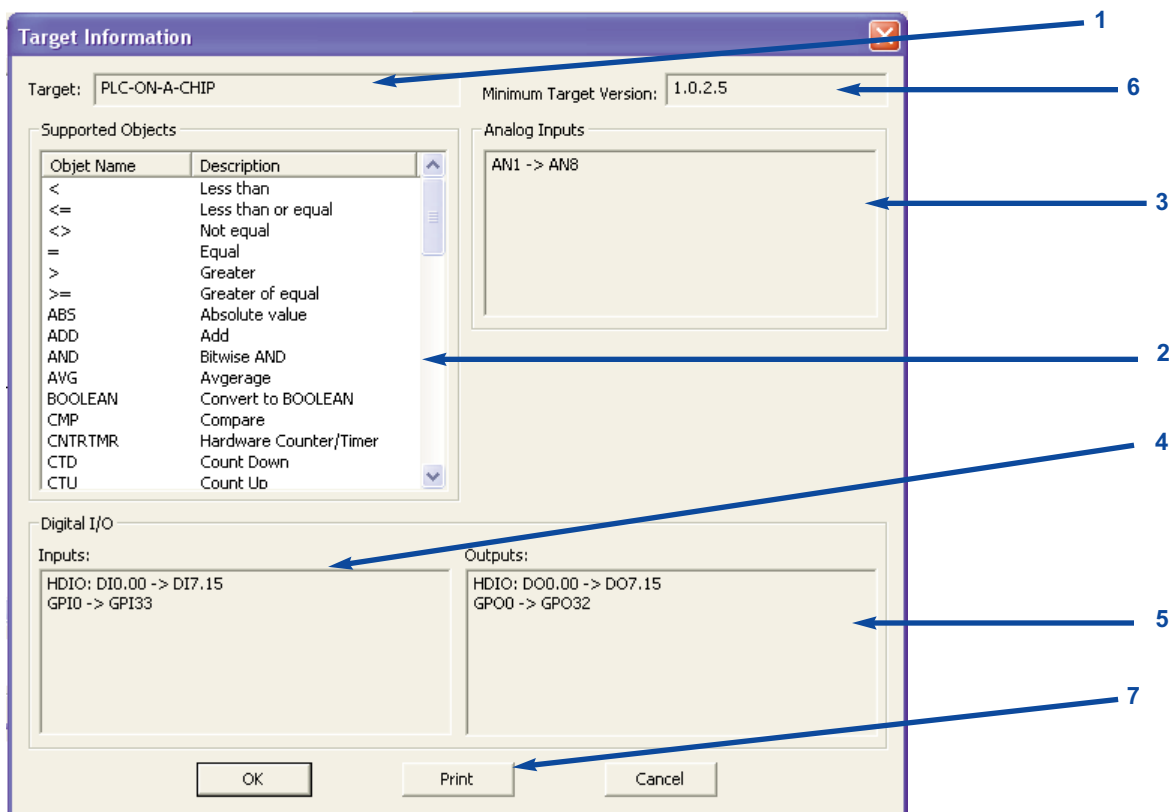


Figure 4.5

- |                           |   |
|---------------------------|---|
| 1. Target:                | The project's selected target.  |
| 2. Supported Objects:     | List and description of supported objects for the selected target.                                    |
| 3. Analog Inputs:         | Supported Analog Inputs.  |
| 4. Digital I/O - Inputs:  | List of supported addressing for digital inputs.  |
| 5. Digital I/O - Outputs: | List of supported addressing for digital outputs.   |
| 6. Minimum Target Version | This is the minimum hardware target version required to be compatible with this version of EZ LADDER. |
| 7. Print                  | Allows printing of the Target Information.  |

## Updating Target Software

As new functions and features to EZ LADDER and new versions of EZ LADDER are developed and released, it will be necessary to “update” the actual target (kernel) with new software to take advantage of the new versions and features.

EZ LADDER provides an easy way to update the kernel on the target.

1. Obtain the new kernel for the target (provided by Divelbiss via CD, email or download).
2. Start EZ LADDER and open any project that uses the target or create a new project with the target.
3. Enter the Monitor Mode.
4. From the menu, select PROJECT....BOOTLOADER.
5. EZ LADDER will connect to the target and the Bootloader dialog will open showing the current version of the target.
6. Click the ERASE USER PROGRAM button to erase the ladder program on the chip. This is recommended before updating the kernel. (This applies to Bootloader versions 1.0.0.5 and later.)
7. Click BROWSE and select the kernel file. The dialog will update showing the selected kernel file version.
8. To update the version, click UPDATE TARGET. A “downloading firmware” status box will appear. The new kernel version is being installed. This may take several minutes. See Figure 4.6.

When completed, all the open dialogs will close and return to just the Monitor Mode. The kernel update is complete.

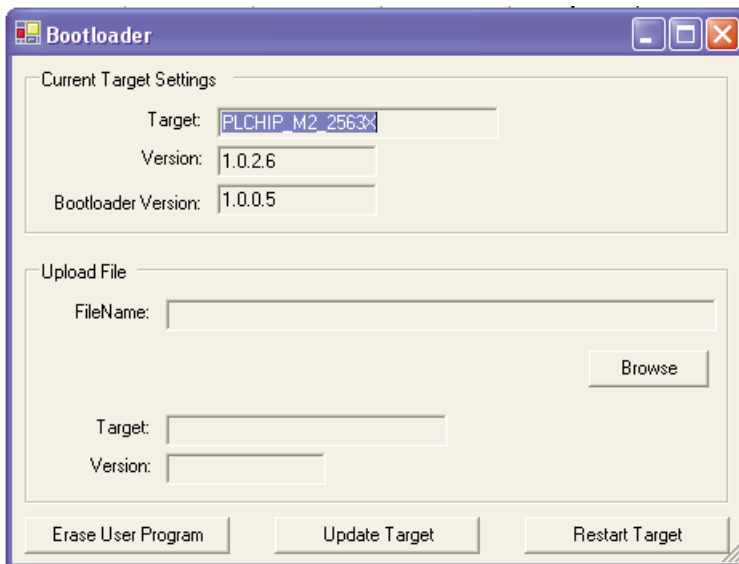




Figure 4.6

 When updating the kernel, DO NOT REMOVE the CABLE or the POWER. If interrupted during this process, the target will be corrupted and return to a “bootloader” mode. Start from the beginning and update the kernel again to correct the problem.

 Only the correct target’s kernel may be installed into a target. The target is checked against the kernel automatically and will display an error if the wrong kernel is selected and an update is attempted. **If a wrong kernel is loaded, contact Divelbiss Technical Support for help regarding removing incorrect kernels.**

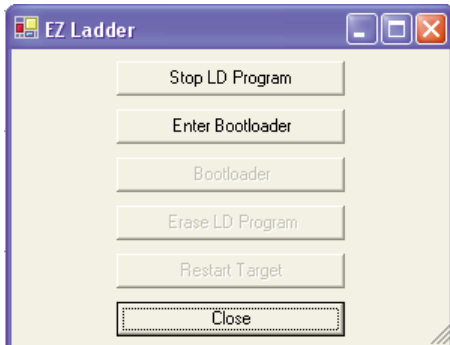
## Target Utilities

Target utilities are provided to help with basic operations and to help correct target problems.

### ***When unable to connect to the target:***

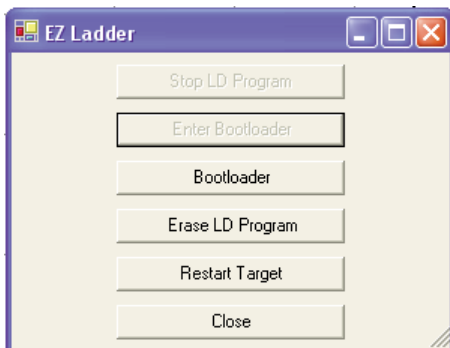
 Please note, these features are available only on targets with a bootloader version of 1.0.0.5 or newer.

When not connected to a target (a project must be loaded), press the F11 key. The 'EZ Ladder' Dialog will appear (see Figure 4.7).



**Figure 4.7**

Disconnect the power from the target and click the ENTER BOOTLOADER. A timing dialog will appear. Connect power to the target during this time. The target will now allow other bootloader operations. (See Figure 4.8).



**Figure 4.8**

**Bootloader** : This opens the 'Bootloader' dialog. From this dialog, the user program may be erased, the target kernel may be updated and the target reset.

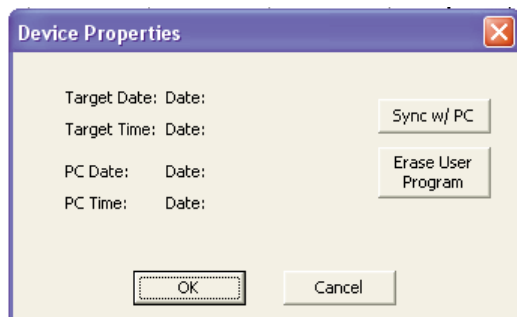
**Erase LD Program** : This erases the ladder diagram program. In the event the program is hanging and preventing a normal connection, this will erase the program to allow a normal connect.

**Reset Target** : This causes a soft restart of the target kernel. This is required when all other bootloader actions have been completed. Without the restart, the kernel will still not connect normally.

**Close** : This closes the EZ Ladder window.

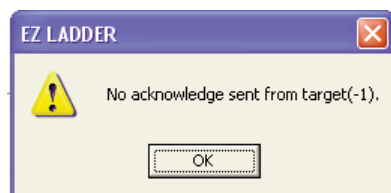
***When able to connect to the target:***

When connected to the target, pressing the F11 key opens the 'Device Properties' dialog. This dialog allows the synchronization of the target's real time clock with the PC clock. The user program may also be erased. See Figure 4.9.



**Figure 4.9**

If the target is not currently running a program with the real time clock, you may receive the following error dialog (Figure 4.10). Click OK to continue.



**Figure 4.10**

# SECTION 5

## USING EZ LADDER TO CREATE LADDER DIAGRAMS



## Creating Ladder Diagram Projects

The first step in creating a ladder diagram project is to open EZ LADDER. EZ LADDER automatically creates a new workspace area as it opens. Figure 5.1 shows recently opened EZ LADDER with a new project window. A new project window may be opened by choosing the **FILE** menu and selecting **NEW**.

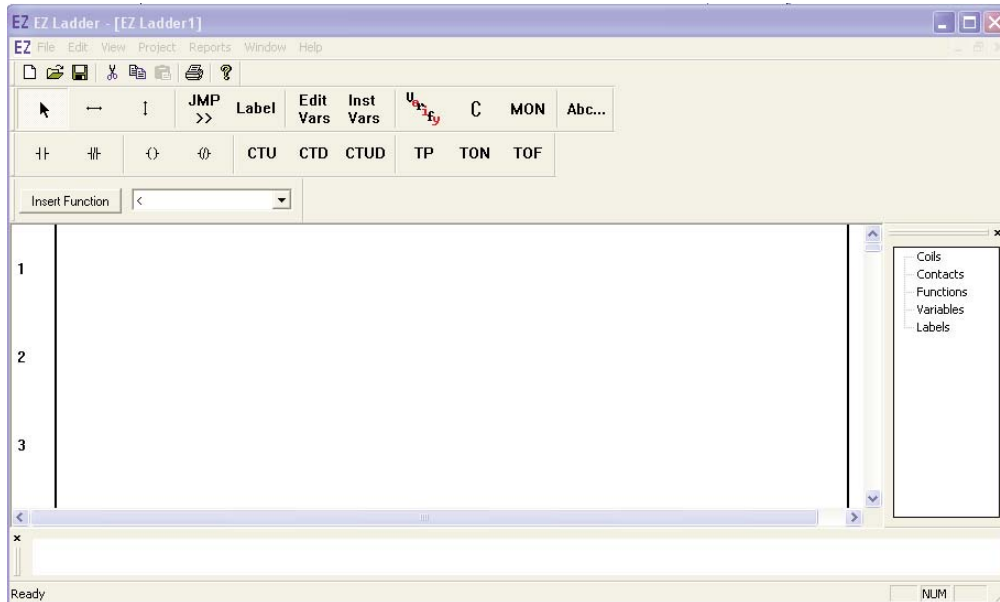


Figure 5.1

The project target settings should be configured before proceeding to place any objects. Use the **PROJECTS** menu and choose **SETTINGS**. Configure the target settings for the hardware that is to be used. EZ LADDER requires the target be selected before any objects can be placed on the workspace. See section 4-*Configuring EZ Ladder for Targets* for more details.

Using the menus and buttons, place ladder diagram objects into the workspace to create a ladder diagram. Later in this section, details will be given on placing variables and objects. Figure 5.2 shows a simple ladder diagram project.

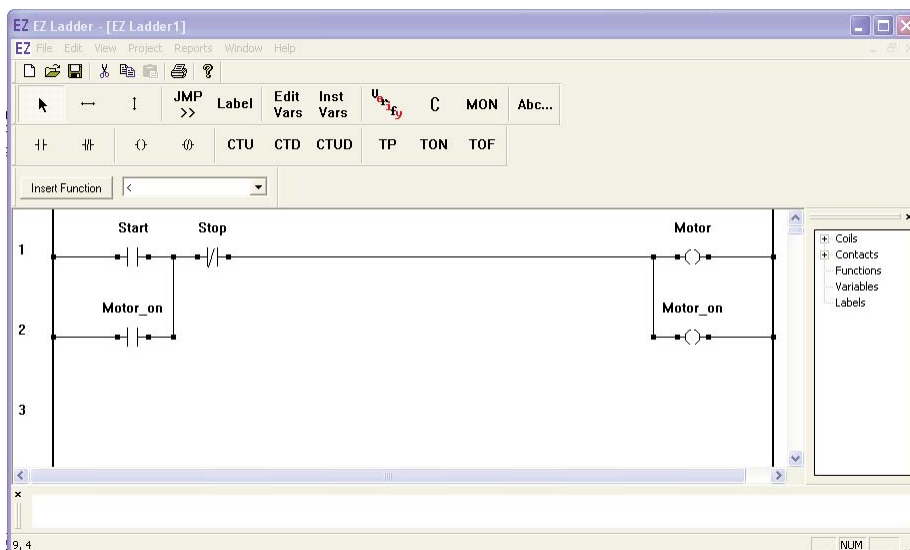


Figure 5.2

## Understanding Objects and Variables

A variable is something that can assume any number of values. Variables are given names for unique identification. Variables can be anything from numbers (1, 2, 3...4.5, 5.6) to text based (ABCDEF).

Variables are an important part of understanding how EZ LADDER uses functions and objects. Each object will require at least one variable to operate properly (either the object uses a variable directly or variables are connected to it using links). The variable types are dependent upon the type of hardware target and the function being placed or used.

Objects (functions) use these variables to “pass” and store changeable values in the ladder diagram project. Figure 5.3. shows a simple ladder diagram with direct and linked variables.

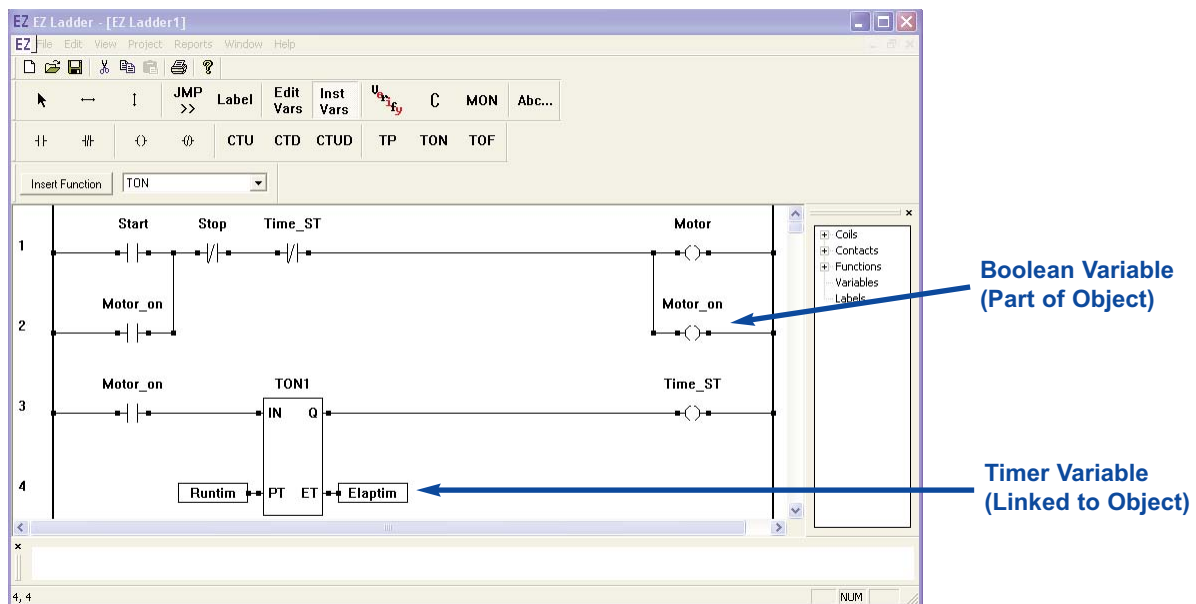


Figure 5.3

## Declaring and Placing Variables

Placing and declaring variables is easy. When placing certain objects (coils and contacts), Contact Properties dialog box will appear. You can choose a variable from the drop-down list or type in a new name and click **OK**. If you select a name that already exists, the object placement is finished. If a new variable name is given, a warning is given that the “*Variable doesn't exist. Create it now?*” Click **YES** to create the variable and configure its type (see later this section on variable attributes). Figure 5.4 shows the Contact Properties dialog box. Figure 5.5 shows the warning dialog. Figure 5.6 shows configuring the variable.

Variables must begin with a letter.

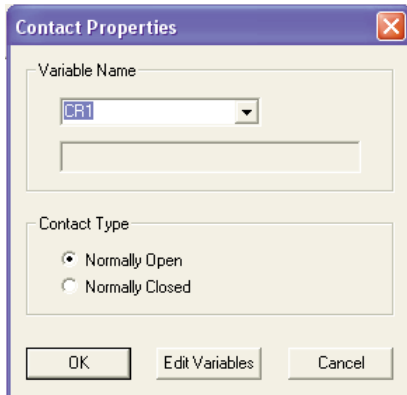


Figure 5.4

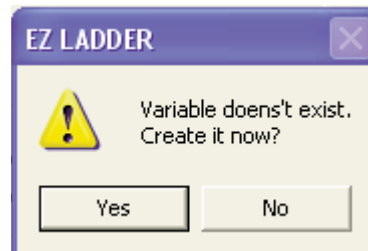


Figure 5.5

When placing other types of objects (that require linked variables), the variables must be inserted manually.

To insert a variable and link to other objects, select the insert variables button from the tool bar **Inst Vars** and position the pointer in the project where the variable is to be inserted. Left-Click and the *Variables* dialog box will open. Select the type of variable you wish to insert using the tabs at the top of the dialog box. From here, if a variable already exists, it can be selected and inserted into the project by highlighting the variable and click **OK**. Figure 5.6 shows the variables dialog box and selecting an existing variable.

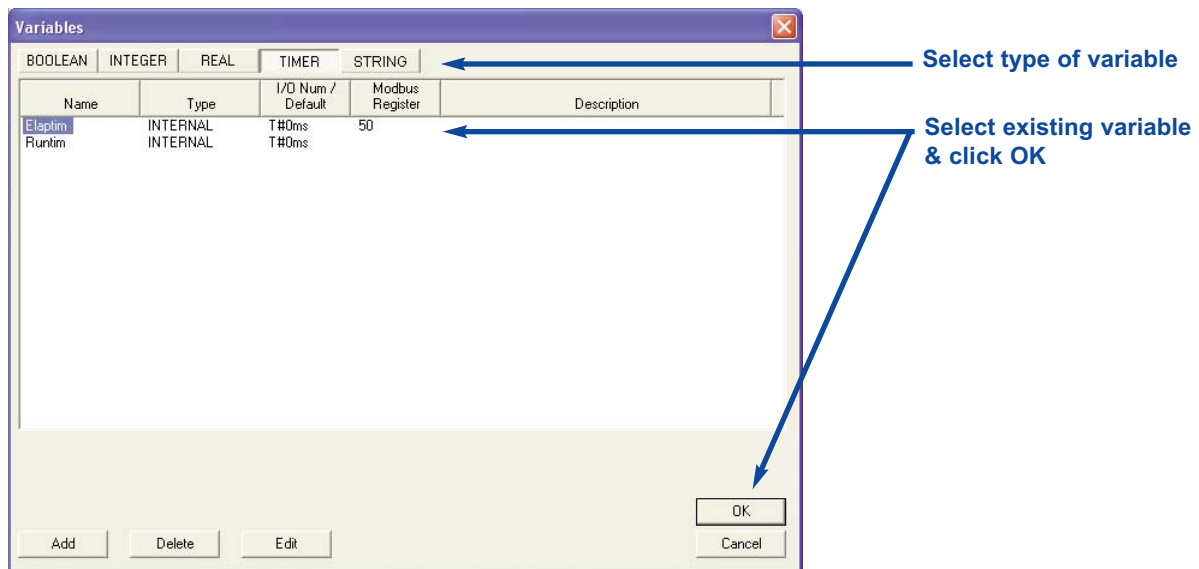


Figure 5.6

If the variable does not exist, it must be added before it can be inserted into the project. Select the variable type to add (see later this section on variable types). Click the **ADD** button. The *ADD Variable* dialog box will open. Complete the required sections (see later this section on variable attributes) and click **OK**. This will show the new variable in the list of the Variables dialog box. Select the variable you just added and click **OK**. The variable is now inserted into the ladder diagram project. Figure 5.7 shows the *ADD Variable* dialog Box.

**NOTE:** When inserting variables for functions, only the type of supported variable for that function will be allowed to be added.

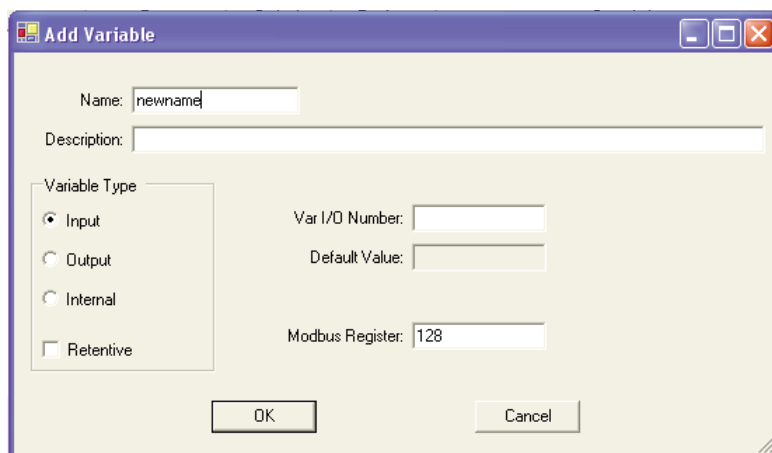


Figure 5.7

## Variable Types

There are five basic variable types (target dependent). Boolean (BOOL), REAL, INTEGER, TIMER, and STRING.

Boolean variables are based on true/false (on/off). The object which uses a boolean variable will either be true or false. When a coil or contact is placed, it is directly tied to a Boolean variable name. Booleans can simply be considered either a 0 or 1 (false or true).

Real variables are based on numbers that use floating point math (use decimal points). Real variables can be ranged from  $-1.7 \times 10^{38}$  to  $1.7 \times 10^{38}$ . Reals are typically used for calculations and with functions where decimal point accuracy is required.

Integer variables are based on whole numbers (no decimal points) Integers can be ranged from -2147483647 to 214483647. Integers are used when decimal points are not required.

String variables are used to represent non-numerical data such as characters to display or transmit serially.

Examples of Variables:

Boolean:	0 or 1, False or True, Off or On
Real:	234.56, 192.345
Integer:	1, 525, 1034
Timer:	Days, Hours, Minutes, Seconds, Milliseconds
String:	This is a string.

## Variable Attributes

When adding new variables, each variable will have attributes that will need to be configured. Attributes are different for different types of variables.

### Integer / Real / Boolean Variable Attributes

The integer, real and boolean variable types share the same required properties.

1. Name: The variable "name" is entered in this field. This name will be used to identify the variable and will be the name viewed in the workspace and any cross reference and reports.
2. Variable Description: This is a text based description of the variable and what it is used for.
3. Variable Type: The variable *type* is selected in this box.  
Select **INPUT** if the variable represents a physical I/O input.  
Select **OUTPUT** if the variable represents a physical I/O output.  
Select **INTERNAL** if the variable has no real world connection but is internal program usage only.  
Retentive check-box is used to specify this variable to maintain its value in the event of a power loss on the target.
4. Var I/O Number: If the variable type is INPUT or OUTPUT type, then the actual I/O point assignment number is entered here (Digital and Analog I/O). See the hardware target manual information for more detail on the I/O assignments.
5. Default Value: The default (power up) value of an *Internal* type variable is set here.
6. Address Register: This is where the variable is assigned a modbus or SPI register (if supported by the target). This is only shown if the target supports modbus or SPI.

*Note: For the Variable I/O Number and the Default Value; only one field will be active to enter data into. The active field will depend on the type of variable selected.*

Figure 5.8 shows the ADD dialog for an integer, real or boolean variable.

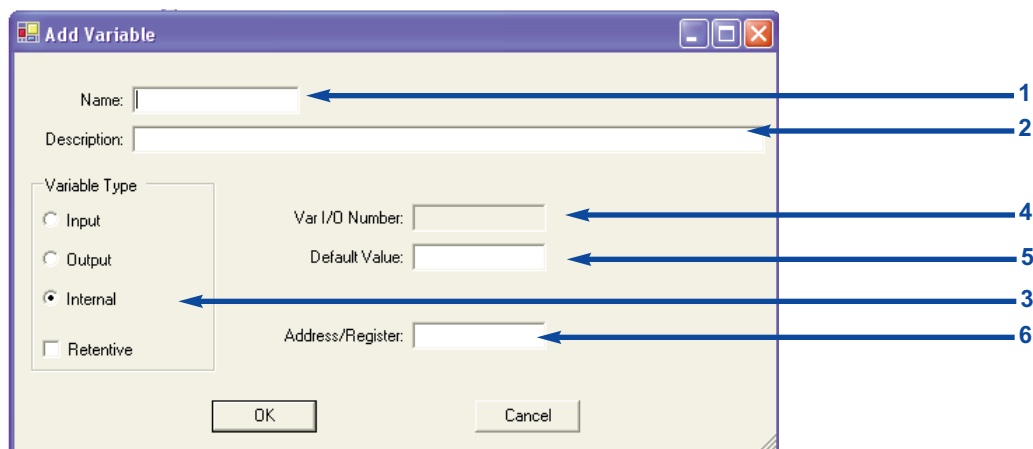


Figure 5.8

Click **OK** to close the *Add Variable* box once all the attributes are complete. Click **OK** again to close the *Variables* dialog box. A real, boolean or integer variable has been successfully added.

### Timer Variable Attributes

Timer variables are required to be used as variables on timer objects for ET (Elapsed Time) and PT (Preset Time). When a timer variable is placed, a different dialog box appears for setting the timer parameters. Figure 5.9 shows the timer dialog box.

1. Variable Name: The variable “name” is entered in this field. This name will be used to identify the variable and will be the name viewed in the workspace and any cross reference and reports.
2. Variable Description: This is a text based description of the variable and what it is used for.
3. Days: The number of days duration the timer is required to function.
4. Hours: The number of hours duration the timer is required to function.
5. Minutes: The number of minutes duration the timer is required to function.
6. Seconds: The number of seconds duration the timer is required to function.
7. Milli-Seconds: The number of milli-seconds duration the timer is required to function. The milli-second resolution is target specific.
8. Retentive: This checkbox specifies the timer to keep its value when power is lost on the target.

Click **OK** to close the *Add Variable* box once all the attributes are complete. Click **OK** again to close the *Variables* dialog box. A timer variable has been successfully added.



Larger times may be entered into fields provided that the total timer value does not exceed 24 days. For example, 1000 ms may be entered and will be considered 1 second when the program executes. However, if 750 hours is entered, its time is greater than 24 days and the timer will not function correctly.

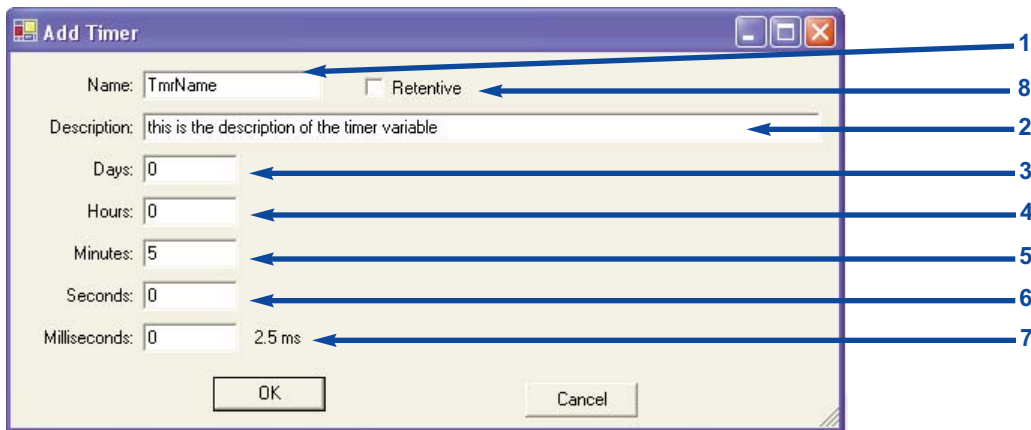


Figure 5.9

### Bit Addressable Variables

Variables may be defined as bit addressable (Integer only). These variables individual bits may be modified (set) or read using other boolean variables. Each integer addressable variable has 32 bits (bit numbers 0-31), each representing the ‘binary’ bit of the total integer number.

### To Set the Bit of a Variable

To set a bit of a bit addressable variable, first create the variable that will be addressed (integer). Next, create an output variable and specify the name and bit to modify. See Figure 5.14.

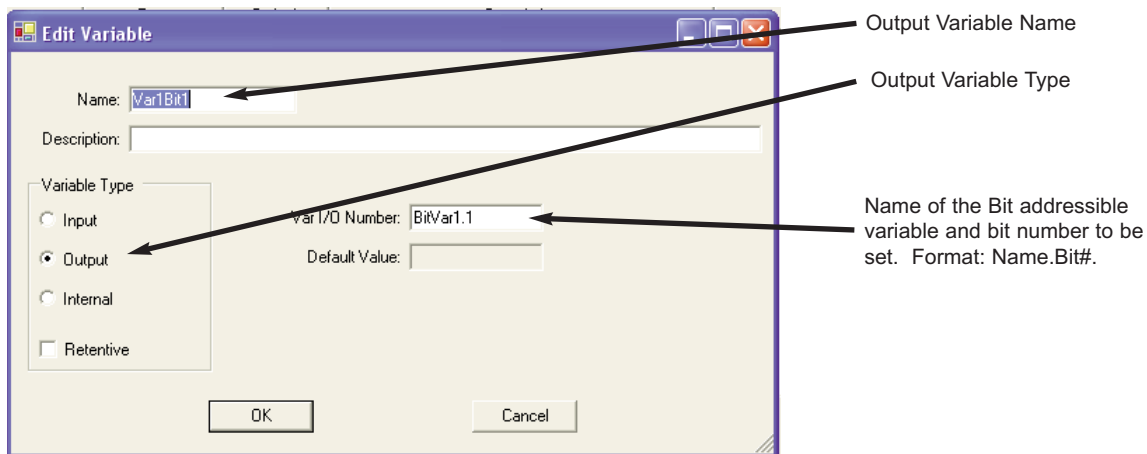


Figure 5.14

When this output is true, the variable bit associated will be true. In this example the "2" bit would be true causing the actual variable BitVar1 to be equal to "2" (plus any additional true bits).

### To Read the Bit of a Variable

To read a bit of a bit addressable variable, first create the variable that will be addressed, if not it does not already exist (integer). Next, create an input variable and specify the name and bit to modify. See Figure 5.15.

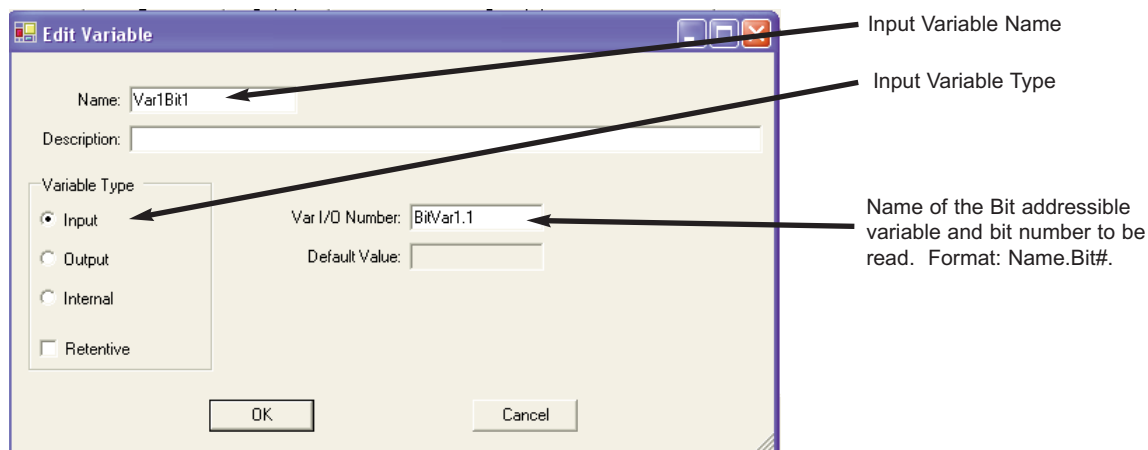


Figure 5.15


When this input is true, it signifies the variable bit associated is true. In this example if the "2" bit would be true, then the input would be true.

## Keeping Variable Values on Power Loss

In the event of a power loss to the target, EZ LADDER is designed to allow ladder diagram variables to be stored and then be reloaded when power is restored. This is called the Retentive feature. See section 8 for more details on the retentive feature.

## Placing Objects and Drawing Links

A ladder diagram is made of objects tied together by links. To place an object in a project, select the object from the tool bar or the tool bar's drop down menu. Position the pointer where the object is to be inserted and left-click. This "places" the object. Repeat this step for other objects as required. Figure 5.10 shows placement of a direct contact and a direct coil. Refer to section 10 for details on all the functions and their variable requirements.

 The last placed object stays selected until a different object is chosen and can be placed multiple times without the need of re-selecting the object.

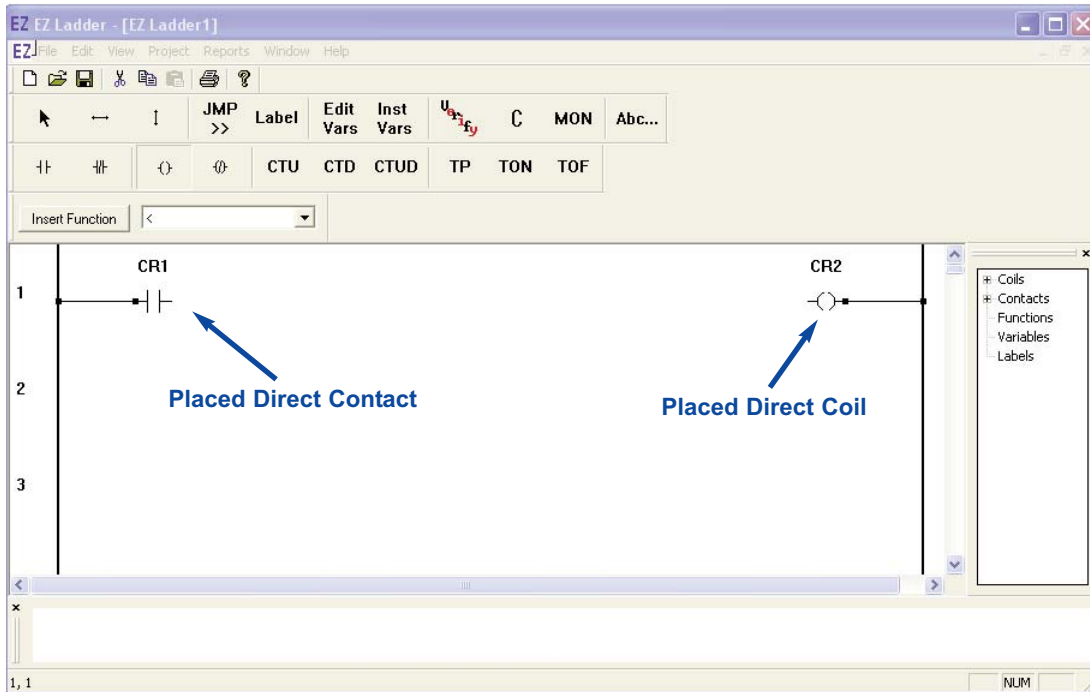







Figure 5.10

 **NOTE:** When placing objects near the left or right power rails, links are automatically drawn to the power rails as shown in Figure 5.10. This also applies when variables are inserted next to functions; the links are automatically drawn from the inserted variable to the function.

All that is required is to draw the link from the input to the output contact. Select the horizontal link tool  from the tool bar. Create the link by clicking in the position where the link should start and then drag to the position where the link should stop (Start by clicking on the open side of the input contact and drag across to the open side of the output contact). This will draw the link between the two objects.

If a vertical link is required (as in parallel circuits), select the vertical link tool . Create the link by clicking in the position where the link should start and then drag to the position where the link should stop.

 **NOTE:** When drawing links to objects and variables, the link must connect to the object for a complete circuit. If the link does not connect, then an error will occur when Verifying or Compiling the ladder diagram program. Figure 5.11 shows a connected link and a link that is not connected.

 When connecting a variable output of a function block to a variable input of another function block, a variable must be placed between the functions in this link for the functions to operate properly (it will compile regardless).



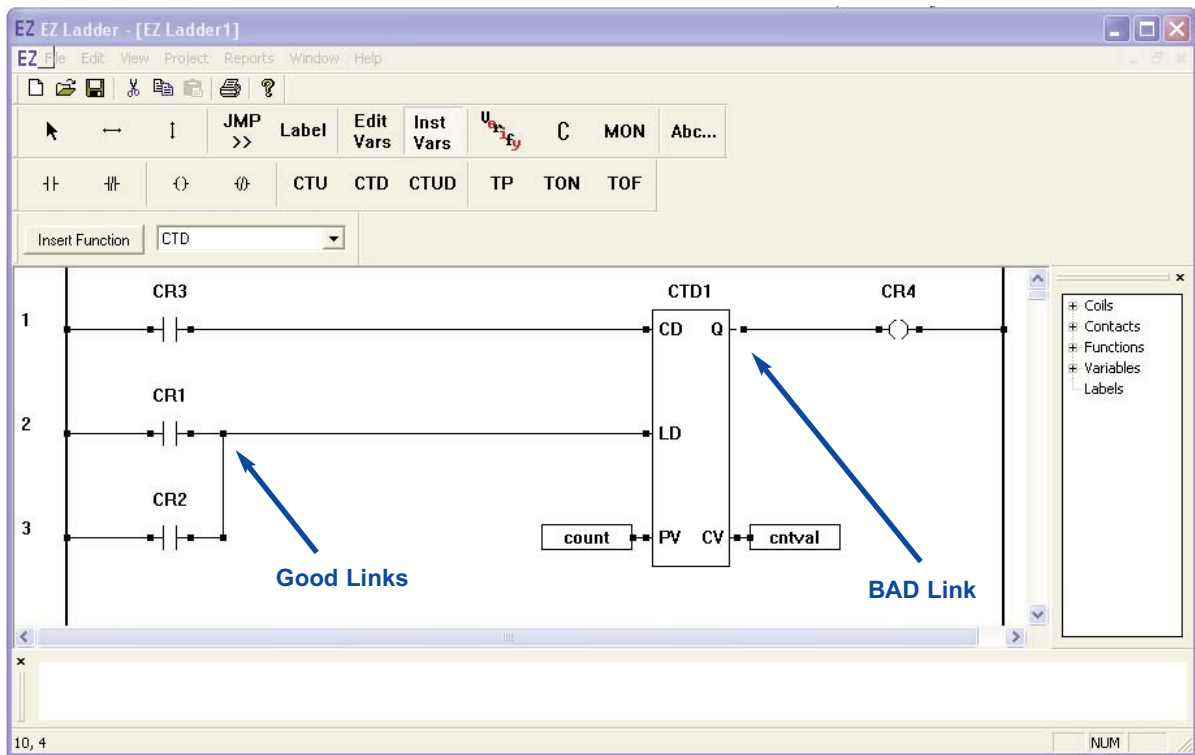



Figure 5.11

## Copy & Paste

Functions, links, parts of rungs and whole rungs may be copied and pasted into another part of the ladder diagram project.


To Copy, choose the Selection Tool from the tool bar . Left-click on the object to copy (for selecting multiple objects or a whole rung, left-click and drag to select what is to be copied). You may also left-click on objects holding the **CTRL** key to select multiple objects.

Once the objects have been selected, from **EDIT** menu, select **COPY** or right-click and select **COPY**.


To paste an object, hover the pointer in the position where you want to paste and right-click. Select **PASTE**.


**NOTE:** When pasting objects or rungs, there must be enough room to paste the copied section or an error will occur. When pasting rungs, move the pointer near the left power rail.

## Inserting and Deleting Rungs

 It is easy to insert and delete rungs in EZ LADDER. To insert a rung, position the pointer where the insertion should be, right-click and select **INSERT RUNG**. To delete a rung, position the pointer on the run to be deleted, right-click and select **DELETE RUNG**.

## Saving Ladder Diagram Projects



Saving a currently open ladder diagram project can be done two different ways. On the Tool Bar, click the  to save the project or on the **FILE** menu, click **SAVE**. If the project has not been named, then a dialog box will appear to enter a name. The **SAVE AS** selection in the **FILE** menu always provides a dialog box for naming the project.

 Beginning with EZ LADDER V1.0.1.0, the file format changed in the ladder diagram. EZ LADDER defaults to the new file format when saving ladder diagrams. Once a ladder diagram has been saved under the new format, it is not compatible with an older version of EZ LADDER. The program can be saved as an older version by changing the file type in the **SAVE AS** dialog (select V1.0.0.X). Saving a ladder diagram as an older file format may cause loss of ladder diagram elements and features (newer features are not available in the older format).

## Verifying and Compiling Ladder Diagrams

After a ladder diagram is created using EZ LADDER, the next step towards having functional hardware is to “compile” the program. Compiling a program is a step where EZ LADDER creates a copy of the actual graphical representation of the ladder diagram and then it is converted into a “compiled” code. This code has no graphical representation, but is now the actual code language for the specific hardware target microprocessor. Generally this type of code is not recognizable or viewable.

During the compilation process, EZ LADDER will perform error checking (missing links, duplicate I/O assignments, etc.). Any errors encountered during the compilation process are displayed in the *Output Window* at the bottom of the workspace (if the view is enabled).

 To compile the program, click the compile  button. The program will *Verify* and attempt to *Compile*. If you have not saved the ladder diagram project, EZ LADDER will require you to do so before compiling. Figure 5.12 shows the previous ladder diagram being verified and displaying an error.

Any errors encountered during the compilation process must be corrected before the compilation will successfully complete and provide operational compiled code. See Appendix A for common error messages. The **VERIFY** button will scan through the program and verify there are no link or object errors. When Compiling, EZ LADDER automatically performs a verify before beginning the compilation process.

Figure 5.13 shows the same ladder diagram with the corrections to the invalid link and compiled successfully. With a successful compile, the ladder diagram project is ready to operate on the hardware target.

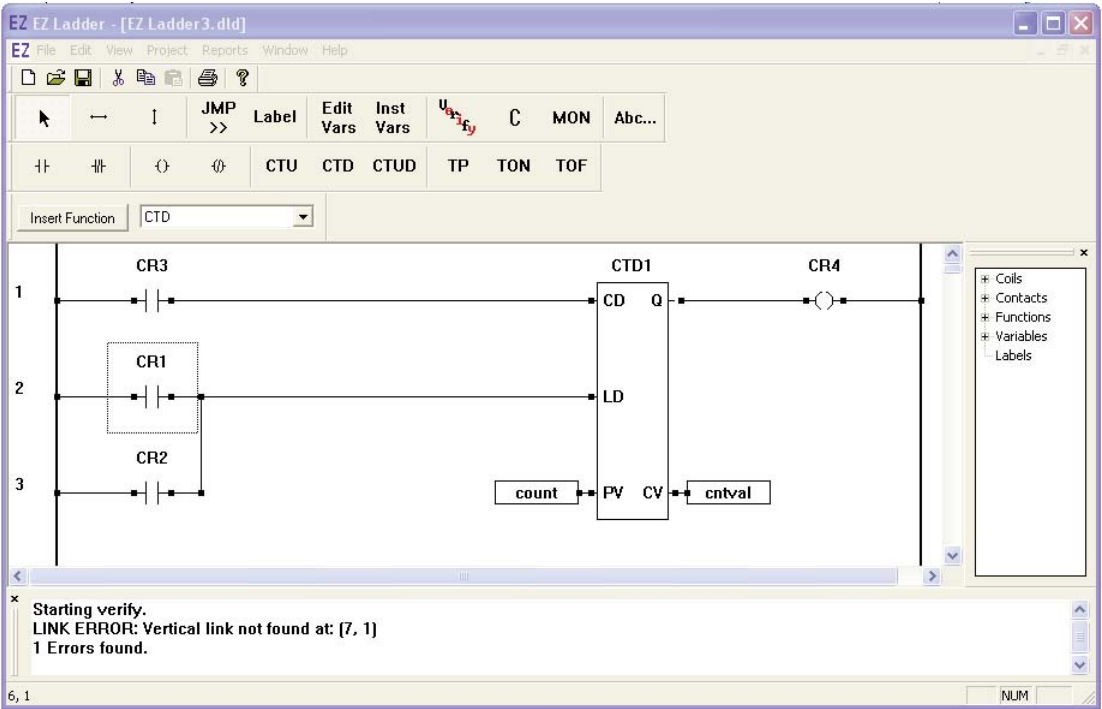


Figure 5.12

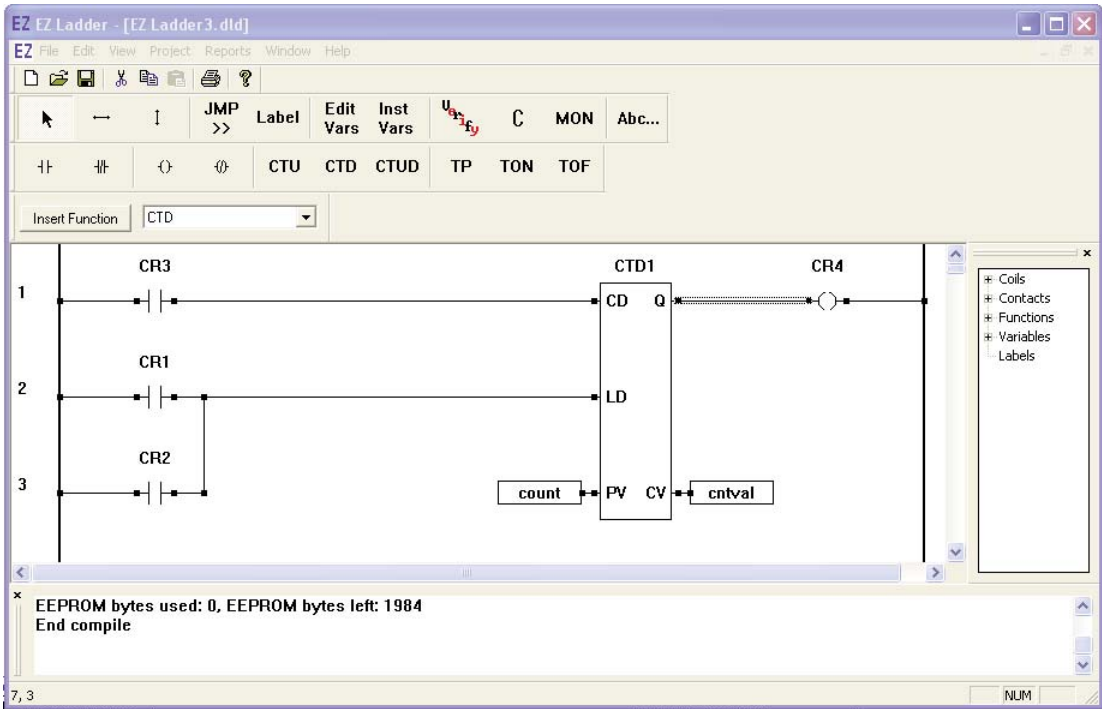


Figure 5.13

# SECTION 6

## DOWNLOADING & MONITORING LADDER DIAGRAMS

## Switching To/From Monitor Mode

EZ LADDER has two modes of operation, Edit and Monitor. The Edit mode is where the ladder diagram is created, edited and compiled. To communicate with hardware targets, EZ LADDER must be placed in the Monitor mode.

To switch to the Monitor Mode, select the Monitor button **MON** from the tool bar. The workspace will change showing different tool bars. Figure 6.1 shows the monitor mode.

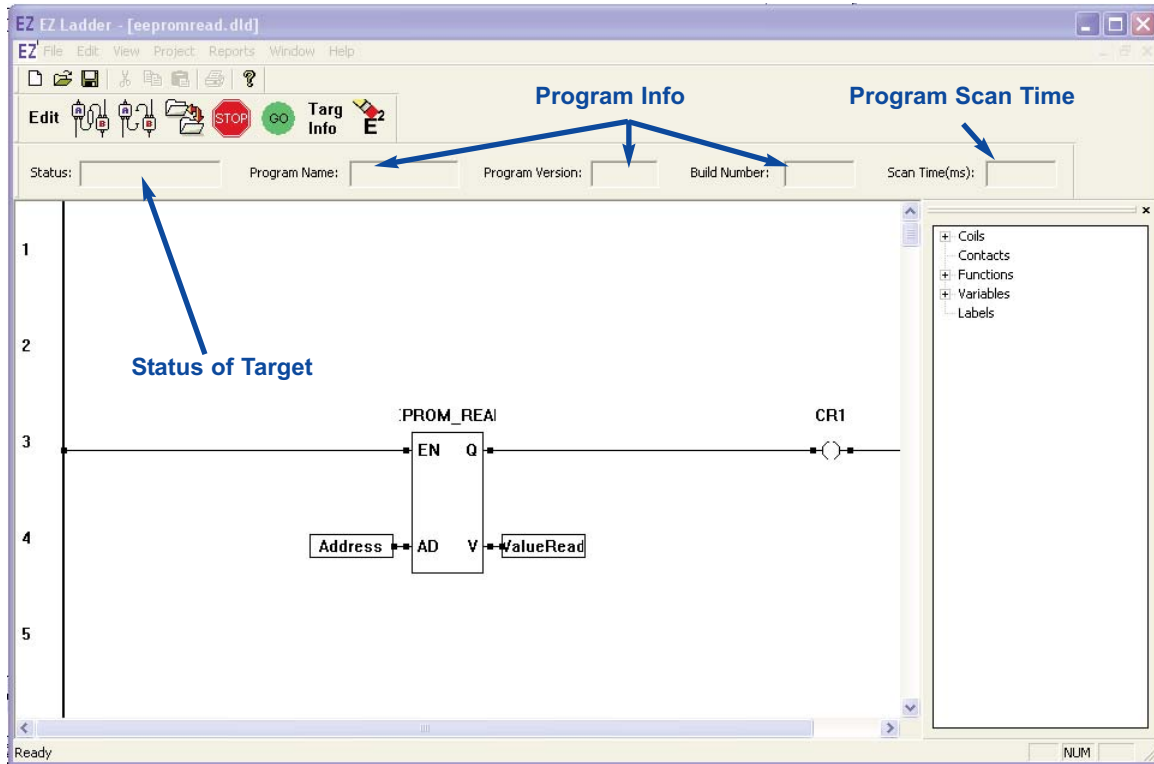


Figure 6.1

The following is a description of the Monitor mode's tool bar buttons.

<b>Edit</b>	<b>EDIT.</b> Exits the Monitor Screen and returns to the EDIT Ladder Diagram Workspace.	<b>CONNECT</b>	Connects the EZ LADDER application to the hardware target for serial communication.	<b>DIS-CONNECT</b>	Dis-connects the EZ LADDER application from the hardware target.
<b>DOWNLOAD</b>	Transfers the active ladder diagram to the hardware target and start its execution.	<b>STOP</b>	Halts execution of a ladder diagram functioning on the hardware target.	<b>START</b>	Starts execution of a stopped ladder diagram on the hardware target.
<b>Targ Info</b>	<b>TARGET INFO.</b> Displays information about the ladder diagram and the hardware target.	<b>ERASE EEPROM</b>	Erases all contents from EEPROM on the hardware target.		

When using the ERASE EEPROM button, use caution as the erase action can not be undone.

To return to the Edit mode, select the Edit button **Edit** from the tool bar. The Monitor mode will close and the Edit mode window will then reappear.

## Connecting to Targets

In order to send a program to a target or monitor the program running on a target, EZ LADDER must first connect to the target.



To connect to the hardware target, select the Connect button from the tool bar. If the opened ladder diagram project in EZ LADDER is already installed on the target (same version), the monitor mode window will show the ladder diagram execution including power flow and scan time (EZ LADDER's status will change to *Running*). If the currently opened ladder diagram project is not loaded, then EZ LADDER's status will change to *Waiting*. If the currently opened ladder diagram project is loaded, but not the same version or build, a dialog box will be displayed with the version information.



To disconnect from the hardware target, select the Disconnect button from the tool bar.

! To monitor a program, the version currently opened in EZ LADDER and the program stored on the target must match name and version. If these do not match, the program must be downloaded to the target or the downloaded version must be opened in EZ LADDER.

! Build information increments each time a program is compiled.

## Downloading to Targets

To monitor a program, the version currently opened in EZ LADDER and the program stored on the target must match name and version. This is accomplished by “downloading” the program to the target. This action sends the compiled program and stores it in the target's non-volatile memory (where it will remain until overwritten).



To download the program to the target; select the Download button from the tool bar. EZ LADDER must be connected to the target before a program can be downloaded. When downloading, a progress indicator dialog box will display the status of the transmission of the program to the target. When downloading is complete, the indicator dialog box will disappear (Target status will be Running and the ladder diagram will be colored for power flow).

When a program is downloaded, it is automatically given the command to execute.



## Power Flow Indications

With a program downloaded and executing, EZ LADDER shows the status of the function and objects by coloring the objects (contacts and coils) and various links. Red indicates power flow (currently energized) while blue represents no power flow (not energized). Figure 6.2 shows a EZ LADDER connected to a target executing a program and is indicating power flow.

## Scan Time

In the Monitor mode, EZ LADDER provides a display of the current scan time for the ladder diagram that is running. The scan time resolution is dependent upon the hardware target. Refer to the hardware target's information for actual scan time resolution.

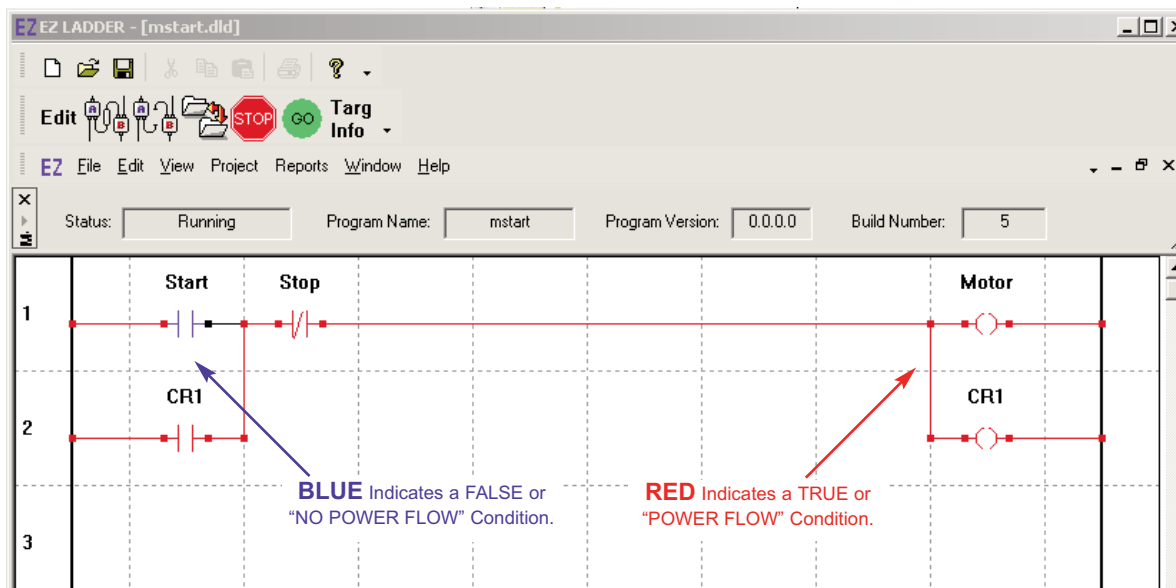


Figure 6.2



The program execution may be stopped by pressing the Stop button  on the tool bar.



To start a stopped program execution press the Go button  on the tool bar.

**Targ  
Info**

To get information about the targets kernel (software it uses to execute the ladder diagram), press the Targ Info button from the tool bar.

## Hover Boxes

In the Monitor mode, EZ LADDER provides additional information in real time. Placing the pointer over an object, will cause a “hover box” to appear. This box provides additional information about the object the pointer is hovering over. Figure 6.3 shows a hover box.

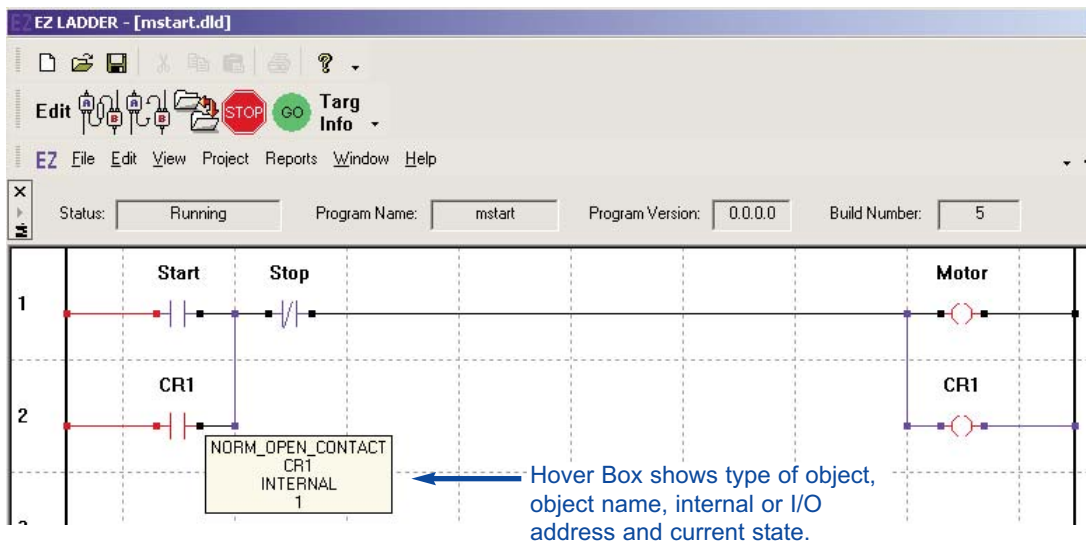


Figure 6.3

### Changing Variable Values



In the Monitor mode, EZ LADDER provides an option for changing the value of a variable while the ladder diagram is executing. Double-click on the object and a dialog box appears with the current state or variable value. This box is editable and the value may be changed. Change the value as needed and click **OK**. The changes take place immediately. The change does not affect the actual ladder diagram (in the Edit mode), only the executing program. This is helpful for adjusting timer and counter values in real time.



Changing a contact variable (boolean) does not always have the desired effect. For example: If the value of an internal coil (that is connected to a real world input) is changed using the dialog box, the actual value will change only until the next scan and then will revert to its real world status. Since all I/O status is re-evaluated each scan, the contacts and coils are updated and will override variable changes. Actual real world inputs cannot be changed at all.



# SECTION 7

## MODBUS SLAVE



## Modbus Slave

Modbus is a register based communication protocol connecting multiple devices to a single network. This network is divided into two types of devices. The first is the “Master”. The master is the device that is in control and initiates communication to the other devices. The other devices are “Slaves”. These devices “listen” for communication from the master and then respond as they are instructed. The master always controls the communication and can communicate to only one or all of the slaves. Slaves can not communicate with each other. EZ LADDER supports Modbus Slave (only on supported hardware targets).

To use the Modbus feature, it must be enabled when selecting the target. From the menu, select **PROJECT... SETTINGS**. The *Project Settings* window will open. Select (highlight) the hardware target. If the target supports Modbus, a **MODBUS** button will become visible and active. Figure 7.1 shows the Project Settings window with a visible **MODBUS** button.

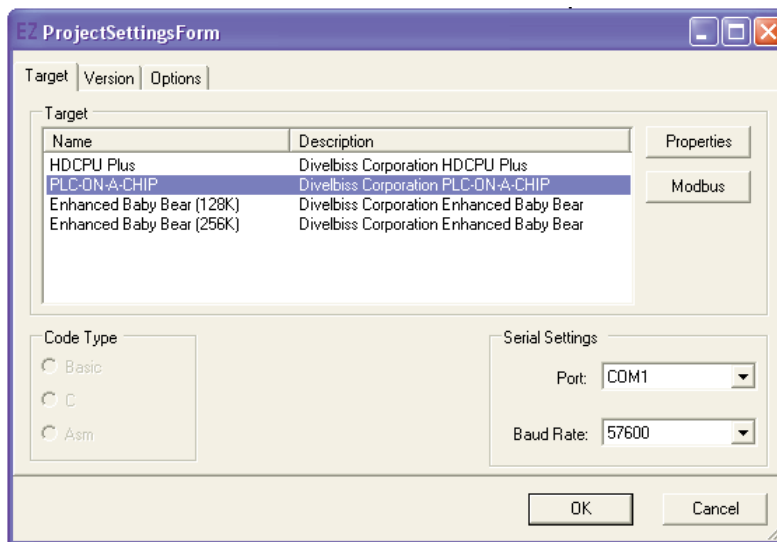


Figure 7.1

## Modbus Configuration

Continue the target configuration (**PROPERTIES** button with some targets) with its I/O, etc. When all the configuration is complete, click the **MODBUS** button. The *Modbus Setup* window will open. To use Modbus, check the *Enable* box. The modbus setup parameter boxes will now become editable. Figure 7.2 shows the *Modbus Setup* window.

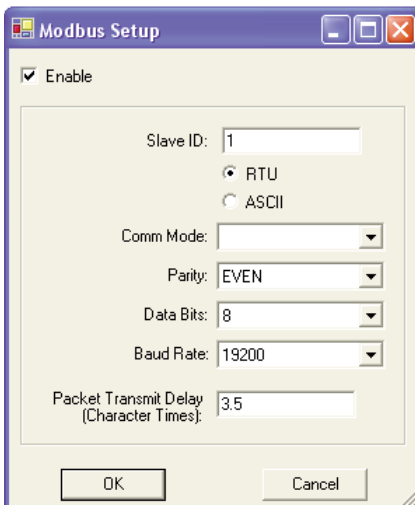



Figure 7.2

Configure the parameters of the Modbus Setup window and click **OK**. The parameters are:

Slave ID:	This is the slave ID (1-247) of the unit where the ladder diagram is to be installed. Each target slave in the network should have its own unique ID number. Duplicate ID numbers on targets will interfere with communications.
RTU:	Binary communication packet (less data and faster than ASCII)
ASCII:	Hex communication packet (more data and slower than RTU)
Comm Mode:	Type of communications RS232, RS422 or RS485.
Parity:	Select parity for the communication packet - Even, Odd or None.
Databits:	Select the number of databits in the communication packet - 7 or 8.
Baud Rate:	Select the baud rate for the serial port - 9600, 19200, 38400, 57600 or 115200 bps.
Packet Transmit Delay:	Minimum number of character times to delay before sending a response. Example: $9600 \text{ baud} - 10 / 9600 * 1000 * 3.5 = 3.65 \text{mSec}$

### Register Assignments

EZ LADDER supports the following register assignments for Modbus communication.

<u>Type</u>	<u>Begin</u>	<u>End</u>	
Coils:	MB_10001 - MB_20000		Used to set state of Output or Internal Coils (variables).
Discrete Inputs:	MB_20001 - MB_30000		Used to read status of Internal or Real World Inputs.
Input Registers:	MB_30001 - MB_40000		Read only registers that the slave can place data into. The master cannot modify these registers.
 Holding Registers:	MB_40001 - MB_50000		Registers that the slave and master may modify. Used to pass data from between the master and slave.

New in V1.0.1.1 or higher, the register number must begin with MB\_.

### Master Functions

EZ LADDER supports functions:

<u>Function</u>	<u>Description</u>
1	Read Coil Status
2	Read Discrete Input Status
3	Read Holding Registers
4	Read Input Registers
5	Write to Single Coil
6	Write to Single Register
15	Write to Multiple Coils
16	Write to Multiple Registers

## Communication Errors

EZ LADDER supports the modbus communication error codes. These codes are reported on the master in the event of a communication error (never on the slave).

If the master attempts to access an invalid register, the master will report “ILLEGAL\_DATA\_ADDRESS” (Error code 2).

EZ LADDER internally tracks the largest register number for each type. If the master attempts to access a register larger than the internal tracked number, the master will report “ILLEGAL\_DATA\_VALUE” (Error code 3).

For more information on master error codes and modbus functions, please refer to the master’s hardware and software information.

## Assigning Registers

Registers are assigned to variables and I/O when the object (coils and contacts) or variable is placed. The *Add Variable* dialog box is where the actual register number is assigned. Figure 7.3 shows the *Add Variable* dialog box.

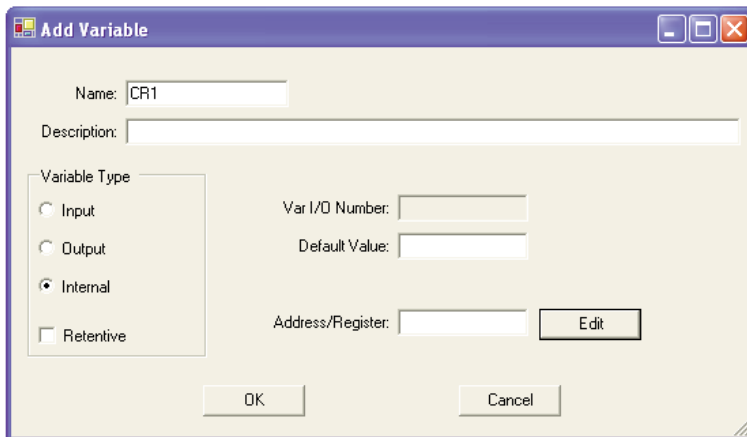


Figure 7.3

To apply a register number to a variable, simply type in the required register address in to the “Address/Register” box or click the **EDIT** button. A new dialog will appear. Using the *Prefix* drop down menu, select the type the Modbus prefix. Using the *Register Type* drop down box, select input or coil. In the Register Index box, type in the actual register number (number only as the ‘MB\_’ will be added automatically). Click **OK**. Figure 7.4 shows the Edit / Address Register dialog box. Figure 7.5 shows a modbus register address.

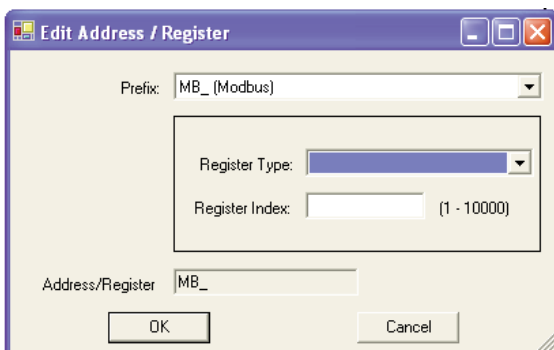


Figure 7.4

**Edit Variable**

Name:

Description:

Variable Type

☐ Input

☐ Output

☒ Internal

☒ Retentive

Var I/O Number:

Default Value:

Modbus Register:

OK Cancel

Figure 7.5

# SECTION 8

## RETENTIVE VARIABLES



## Retentive Variables

Depending upon the hardware target used, EZ LADDER provides the option of assigning variables to be stored in the event power is lost to the target. This feature must be supported by the hardware target.

**Note:** The maximum amount of memory for retentive variables is 100 bytes. During programming, you must take into consideration how many variables are retentive and how much memory they use. Real and Integer variables require 4 bytes each while boolean variables require 2 bytes each.

### How it works

First, variables must be 'declared' as retentive. This is done in the 'Add Variable' or 'Edit Variable' dialog box. To declare the variable as retentive, click the 'Retentive' check box as shown in Figure 8.1.

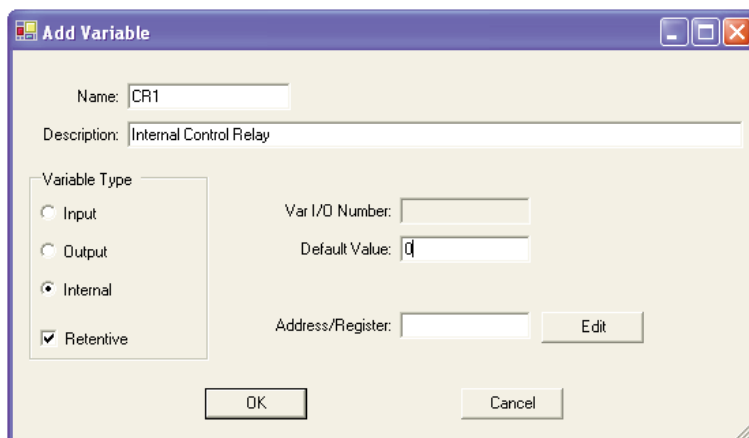


Figure 8.1

When the program is running and a power loss is detected, all variables that are designated 'retentive' are automatically stored into the target's EEPROM.

When power is restored, the target automatically reload from the EEPROM the actual state of the variables on power loss and restores the variables when the program runs.

# SECTION 9

## SSI ENCODER





### SSI (Encoder) Input

For targets that support SSI Encoders, EZ LADDER provides a Graycode SSI interface function block .

For this function to work, the hardware target must support the GC\_SSI block, be 'installed' in the target settings and be connected properly using an interface circuit (if required from the target's SPI port to the encoder (conversion from SPI to differential ). See the target's hardware manual or datasheet for details on this interface.

Some targets such as PLC on a Chip, require the 'installation' of the SSI port before it can be used. If required, click on *PROJECT...SETTINGS*. Select the target and click **PROPERTIES** and the **ADD** button. Select 'SSI' from the list and click **OK**, **OK** and **OK** to exit the Project Settings dialog box. Save the project. Figure 9.1 shows 'installing' the SSI option using the PLC on a Chip target.

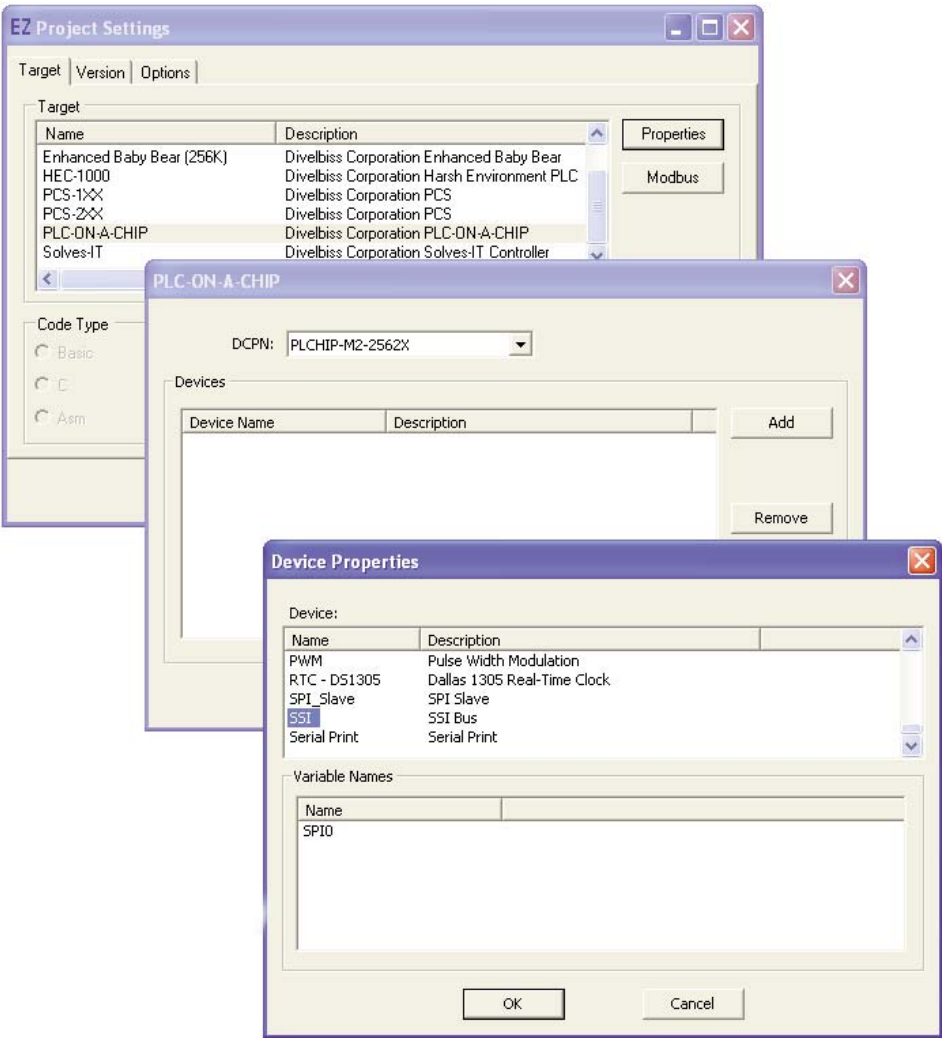


Figure 9.1

To use the GC\_SSI block, select it from the functions list and place in the project workspace. A dialog box will open. See Figure 9.1.

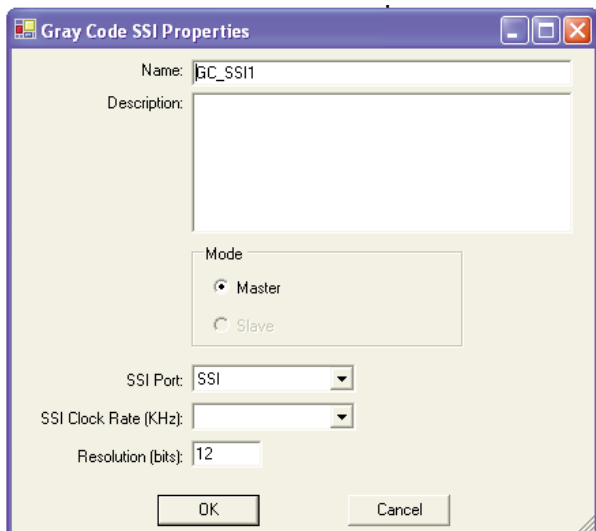


Figure 9.2

Select the SSI Port using the drop down menu (some targets may support multiple GC\_SSI ports). Select the SSI clock Rate using the drop down menu. This rate is in KHZ. Enter the resolution and click **OK**.

When functioning, the GC\_SSI block returns an integer value representing the Graycode reading from the encoder. This is read serially, converted from Graycode to a binary number then returned to the block as an Integer output.

### Slave SSI (Encoder) Input

Two “targets” may be connected to one GC\_SSI encoder to provide redundancy. These two are connected to a single GC\_SSI encoder input and are configured as *Master* and *Slave*.

See Section 20 for more details on the GC\_SSI function block.

# SECTION 10

## PWM OUTPUTS



## PWM Outputs

EZ LADDER provides two functions that will provide control over the target's hardware PWM outputs (if supported on the target). These two functions are PWM and PWM\_FREQ. Refer to this manual's Section 20 - Functions for details on using and configuring these functions.

Before these functions can be used to control the target's hardware PWM outputs, the outputs must be installed using EZ LADDER in the target's configuration.

To install PWM Outputs in the target configuration:

1. Use the menu, select *PROJECT....SETTINGS*. The ProjectSettingsForm dialog will open. Select the target.
2. Click the **PROPERTIES** button.
3. Click the **ADD** button.
4. From the provided ADD dialog *Devices* list, select the PWM (Pulse Width Modulation). See Figure 10.1.

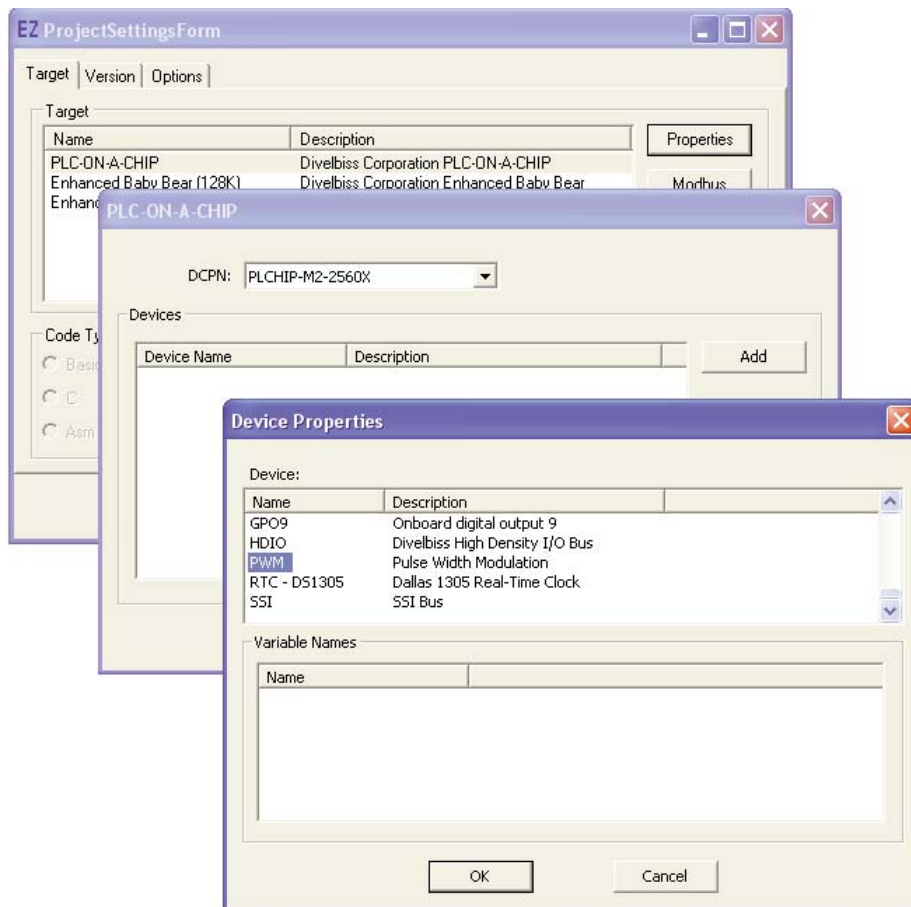
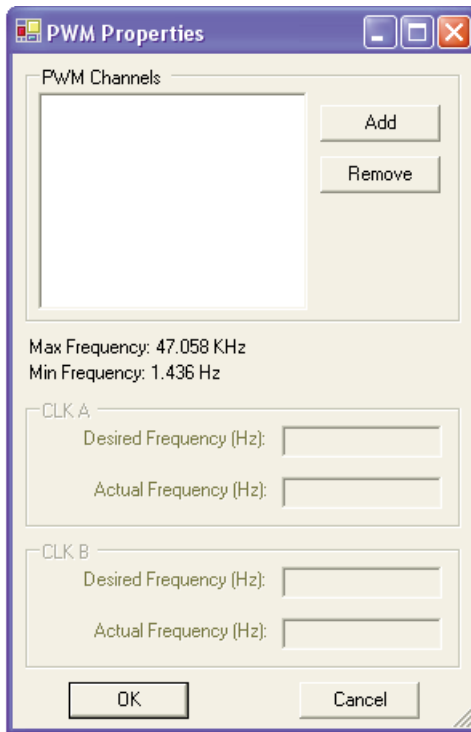


Figure 10.1

5. Click **OK** to install the PWM device and return to the target's properties dialog.
6. Click on PWM and a Properties button will appear on the right side of the target properties dialog.
7. Click the **PROPERTIES** button. The PWM Properties dialog will open. This dialog is used to add, remove and configure the PWM channels on the target. See Figure 10.2.

**Figure 10.2**

8. Click the **ADD** button. The ADD PWM dialog will open. Select the channels you wish to install (use).

Note: The PWM features are target specific. For the remainder of this we will use the PLC on a Chip as the example. See the hardware target information for more details on the hardware PWM availability.

9. Enter the desired frequency for Clock A and Clock B (if installed). The PLC on a Chip has 8 available PWM Channels. These 8 channels are either controlled with Clock A or Clock B. This allows two different PWM frequencies. The Minimum and Maximum frequencies are displayed in the PWM Properties dialog. The frequency for Clock A and Clock B must be in this range. The actual frequency is what will be seen on the actual PWM hardware output channels (as close as possible to the desired frequency; this is due to limitations of the hardware). See Figure 10.3.
10. Click **OK** to close the PWM Properties and save the changes.
11. Click **OK** to close the Target Properties and save the changes.
12. Click **OK** to close the ProjectSettingsForm dialog and save the changes.

The PWM channels are now ready for use using the PWM and PWM\_FREQ functions.

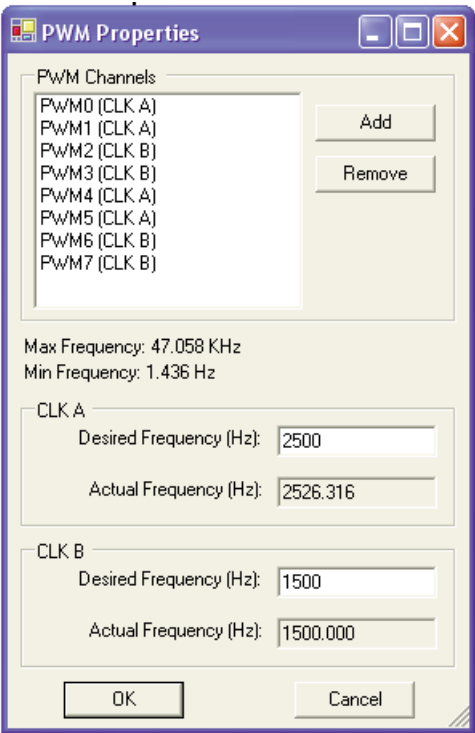


Figure 10.3

# SECTION 11

## SPI SLAVE



## SPI SLAVE

EZ LADDER supports SPI Slave configuration and use depending upon the actual hardware target that is to be used. The following information is necessary to use the SPI Slave feature of EZ LADDER.

### SPI SLAVE CONFIGURATION

Before these functions can be used to control the target's hardware SPI port, it must be installed using EZ LADDER in the target's configuration.

To install SPI Slave in the target configuration:

1. Use the menu, select *PROJECT...SETTINGS*. The ProjectSettingsForm dialog will open. Select the target.
2. Click the **PROPERTIES** button.
3. Click the **ADD** button.
4. From the provided ADD dialog *Devices* list, select the SPI\_Slave. See Figure 11.1.

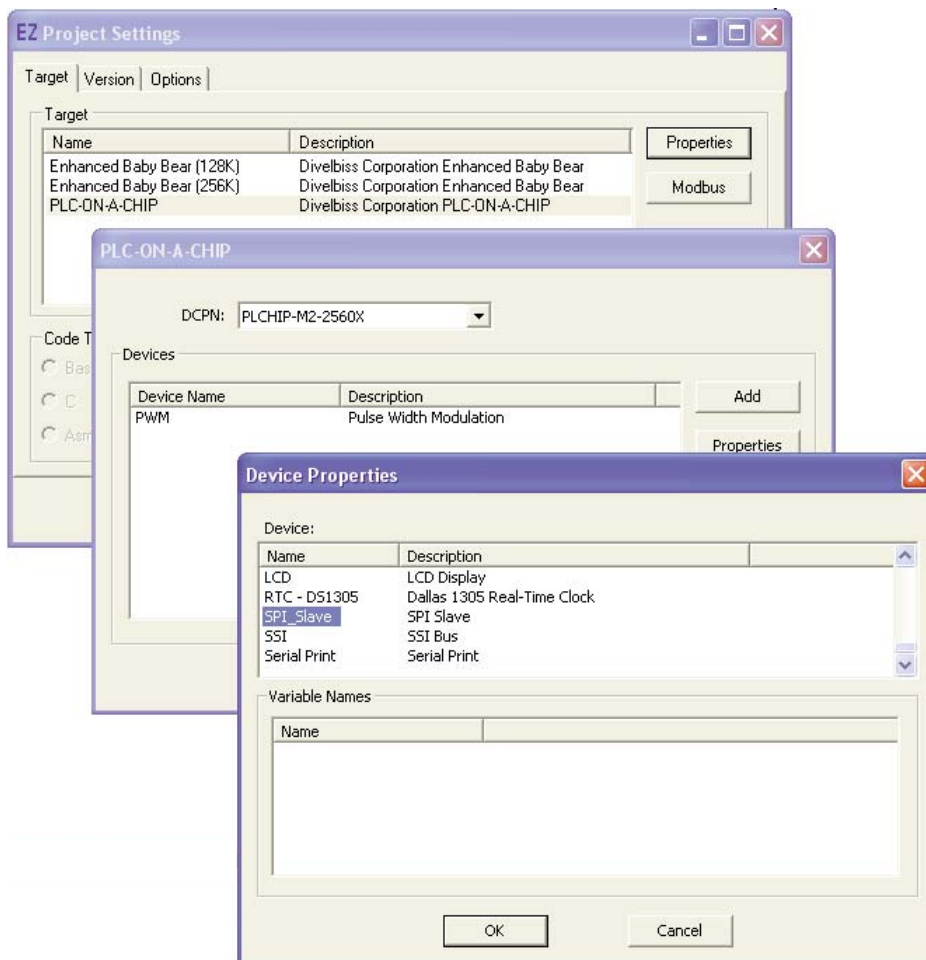


Figure 11.1

5. Click **OK** to install the SPI\_Slave and return to the target's properties dialog.
6. Click on SPI\_Slave and a Properties button will appear on the right side of the target properties dialog.
7. Click the **PROPERTIES** button. The SPI\_Slave Properties dialog will open. This dialog is used to configure the SPI\_Slave on the target. See Figure 11.2.



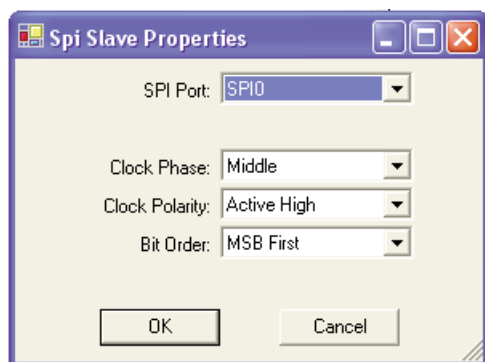


Figure 11.2

8. Select the **SPI Port** to use from the available using the drop-down menu. Note SPI0 is the only port support for SPI Slave.
9. The **Phase Clock** is currently hard-coded to Middle.
10. Select the Appropriate **Clock Polarity** for your application needs, Active High or Active Low using the drop-down menu.
11. Select the Appropriate **Bit Order** for your application needs, MSB First or LSB First using the drop-down menu.
12. Click **OK** to close the SPI\_Slave Properties and save the changes.
13. Click **OK** to close the Target Settings Properties and save the changes.
14. Click **OK** to close the ProjectSettingsForm dialog and save the changes.

The SPI\_Slave is now ready for use.

## USING THE SPI SLAVE FEATURE

The SPI\_Slave feature provides an easy way to “pass” data to an SPI Master device. The data is simply stored in registers that the Master will read. There are a total of 512 registers, each of which are 32-bit. To store data into a register, use the Address Register (add variable / edit variable properties box) box to apply the address of the SPI register that will be assigned to the variable. SPI registers begin with SPI\_.

You may add the SPI Address / Register manually or click the edit button which opens a dialog with drop down menus to help with the entry of the SPI address. See Figure 11.3.

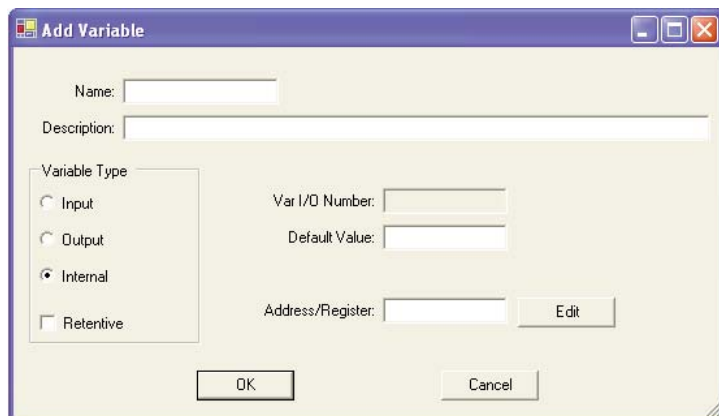


Figure 11.3

## USING THE SPI SLAVE FEATURE Con't

The following is important information regarding the implementation an use of the SPI\_Slave feature in EZ LADDER and PLC on a Chip Targets.

### Register Assignments

SPI Register Beginning Address: 0x0000  
 SPI Register Ending Address: 0x01FF  
 Registers: 32 Bits Each  
 Total # of Registers: 512  
 Naming: SPI\_regnum

### Communications Protocol

**Read / Write Bit:** The high-order bit selects, Read = 0, Write = 1

**Control Word:** 16 Bit Control Word, 32 bit data

**Data Shift:** Shifts most significant byte first

**Chip Select:** Chip select to stay low for byte transfer and MUST go high after each byte for at least 1/2 of the clock cycle.

**Clock Frequency:** Minimum is 10KHz, Maximum is 15KHz.

**Read / Write Delay:** A 1mS delay is required between read and write transfers. This allows time for everything to stabilize and reset after each action.

**Read Command & Data Read Delay:** A 50µS delay is required between sending the read command and actually reading the data.

**Read /Writing Sequential Delay:** A 50µS delay is required between reading and writing sequential data.

**Please Note:** If there is more than 1ms between bytes, then the command is reset and the current byte is treated as the first byte of a new command.

### WRITE:

#### Master

W	-	-	-	-	-	-	A8	A7	A6	A5	A4	A3	A2	A1	A0
---	---	---	---	---	---	---	----	----	----	----	----	----	----	----	----

D 31	D 30	D -	D -	D 11	D 10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
------	------	-----	-----	------	------	----	----	----	----	----	----	----	----	----	----

#### Slave

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

0	0	-	-	0	0	0	0	0	0	0	0	0	0	0	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

### READ:

#### Master

R	-	-	-	-	-	-	A8	A7	A6	A5	A4	A3	A2	A1	A0
---	---	---	---	---	---	---	----	----	----	----	----	----	----	----	----

-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

#### Slave

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

D 31	D 30	D -	D -	D 11	D 10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
------	------	-----	-----	------	------	----	----	----	----	----	----	----	----	----	----

## USING THE SPI SLAVE FEATURE Con't

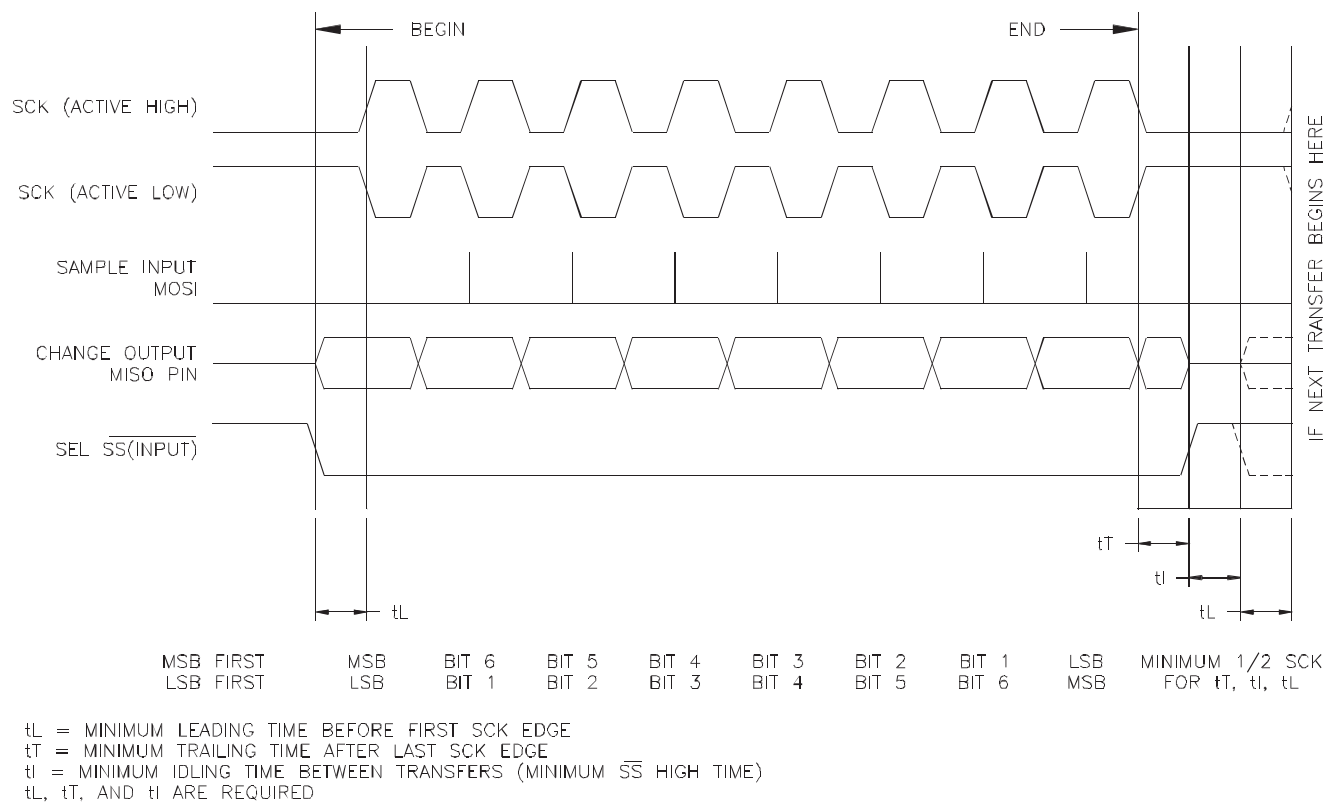
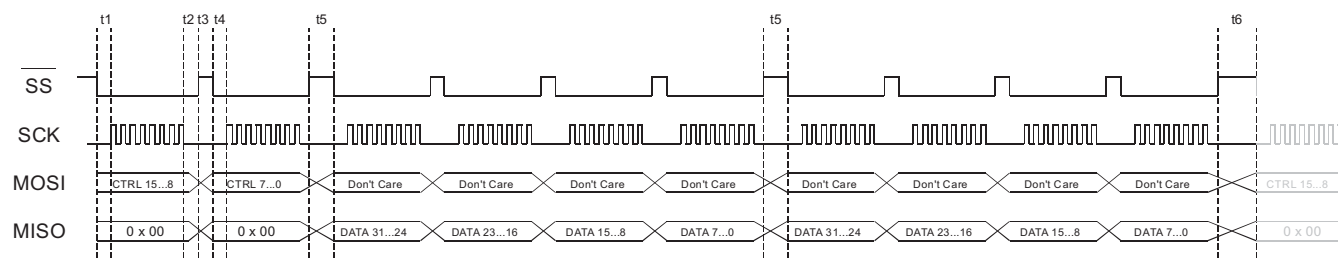


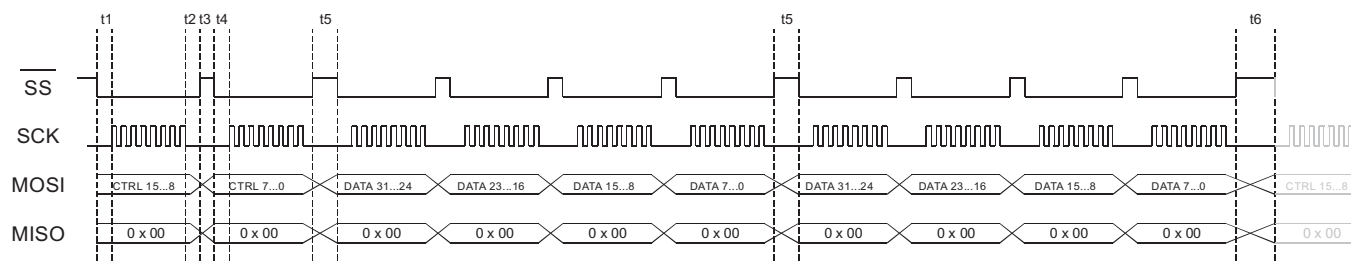
Figure 11.4

## DETAILED COMMUNICATIONS TIMING DIAGRAMS

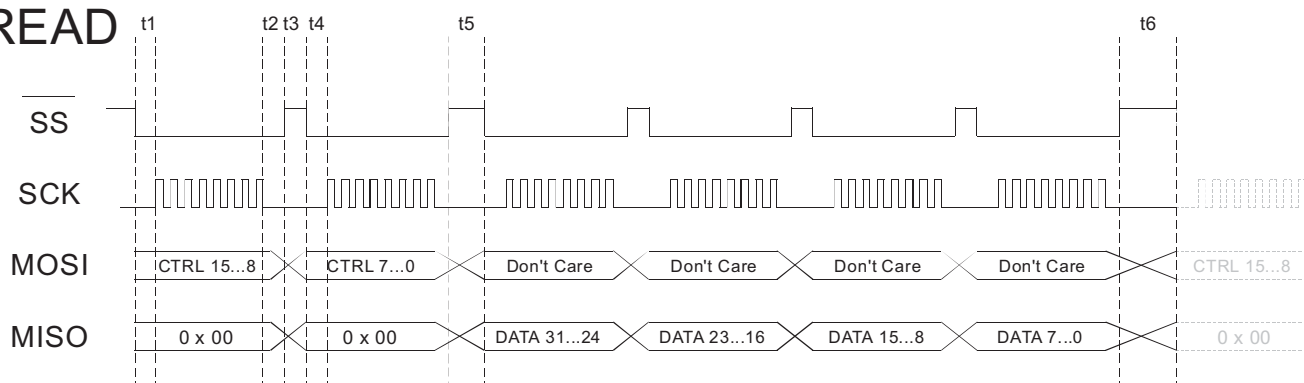
### READ SEQUENTIAL



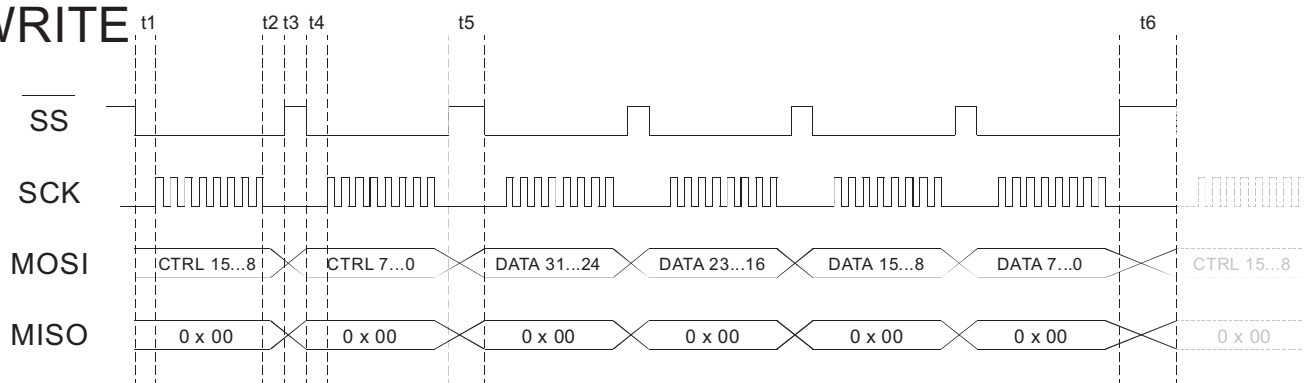
## WRITE SEQUENTIAL



## READ



## WRITE

**Notes:**

1. All Detailed Communications Timing Diagrams are shown using ACTIVE HIGH SCK.
2.  $t_1$ ,  $t_2$ ,  $t_3$  and  $t_4$  = min 1/2 SCK
3.  $t_5$  = min 50 $\mu$ S
4.  $t_6$  = min 1mS

# SECTION 12

## SERIAL PRINT

## Serial Print Functionality

The Serial Print feature provides that ability to serially print using the multi-purpose port on the target. The Serial Print functionality must be 'installed' on the target before it may be used.

Divelbiss standard controllers based on PLC on a Chip (Enhanced Baby Bear, PCS-XXX, etc) are configured based on the part number. To configure, select the correct part number and then check the box for "Enable Serial Print" in the Target's Properties dialog box. When this "Enable Serial Print" box is checked, the SERIAL PROPERTIES button becomes active. Click SERIAL PROPERTIES and jump to step 8 to configure the serial port.

### To install Serial Print in the target configuration for PLC on a Chip:

1. Use the menu, select *PROJECT....SETTINGS*. The ProjectSettingsForm dialog will open. Select the target.
2. Click the **PROPERTIES** button.
3. Click the **ADD** button.
4. From the provided ADD dialog *Devices* list, select the Serial Print. See Figure 12.1.

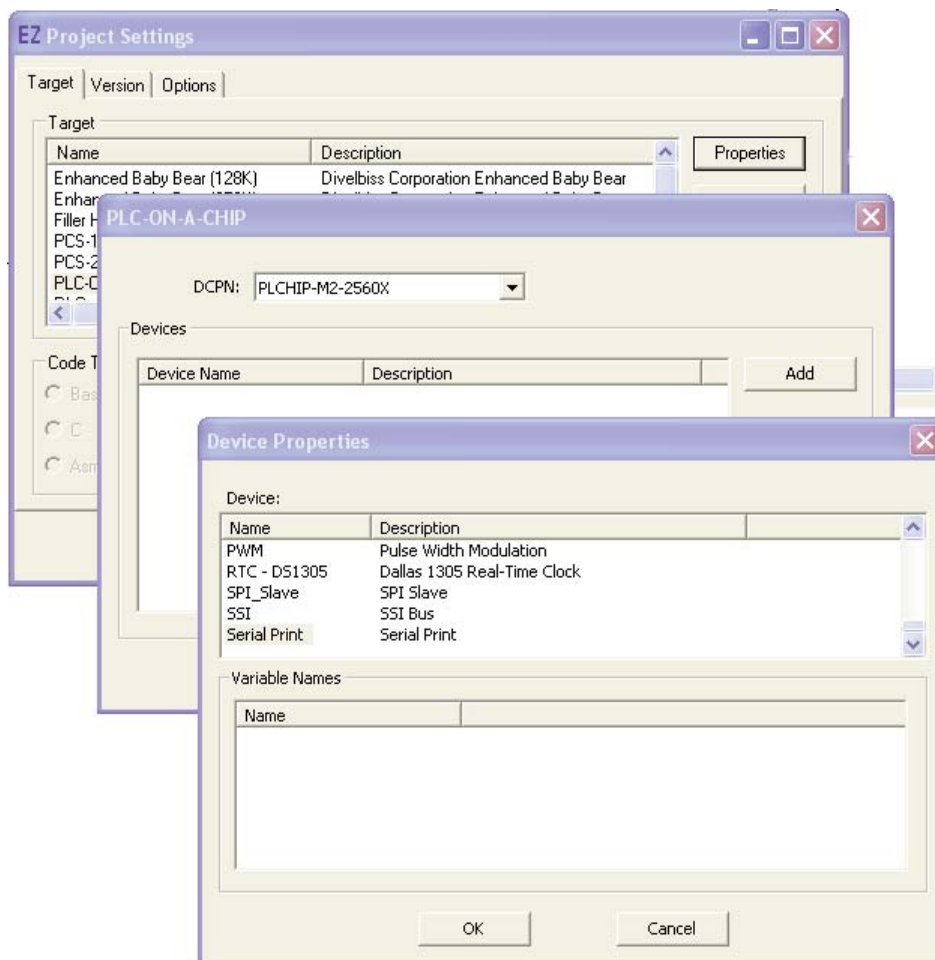


Figure 12.1

5. Click **OK** to install the Serial Print and return to the target's properties dialog.
6. Click on **SERIAL PRINT** and a **Properties** button will appear on the right side of the target properties dialog.
7. Click the **PROPERTIES** button. The Serial Properties dialog will open. This dialog is used to configure the Serial Port on the target. See Figure 12.2.

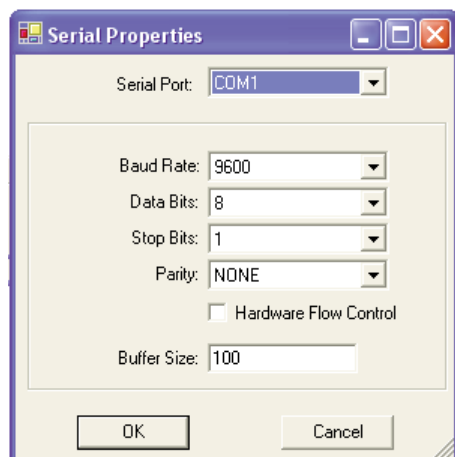


Figure 12.2

8. Select the **COM 1** to use from the available using the drop-down menu. Note COM 1 is the only port support for PLC on a Chip Targets at this time.
9. Select the **Baud Rate**. Supports 9600 to 115.2K baud.
10. Select **8 for Data Bits**. PLC on a Chip Targets currently only support 8 bit data.
11. Select **1 Stop Bit**. PLC on a Chip Targets currently only support 1 stop bit.
12. Select **None for Parity**. PLC on a Chip Targets currently only support None for parity.
13. Check the **Flow Control Box** if you require flow control.
14. Set the **Buffer Size**. It defaults to 100 bytes.
15. Click **ok** to close the Serial Properties and save the changes.
16. Click **ok** to close the Target Settings Properties and save the changes.
17. Click **ok** to close the ProjectSettingsForm dialog and save the changes.

The Serial Print is now ready for use. Refer to Section 20 - EZ LADDER Functions for printing serially using the SERIAL\_PRINT Function.

# SECTION 13

## LCD DISPLAYS





## LCD Functionality

The LCD Display feature provides that ability to add an LCD display and print to it from the ladder diagram. The LCD Display functionality must be 'installed' on the target before it may be used. For information on connections and compatability of the display, please refer to the PLC on a Chip Circuit Design Guidelines document.

To install the LCD display in the target configuration:

1. Use the menu, select *PROJECT....SETTINGS*. The ProjectSettingsForm dialog will open. Select the target.
2. Click the **PROPERTIES** button.
3. Click the **ADD** button.
4. From the provided ADD dialog *Devices* list, select the LCD. See Figure 13.1.

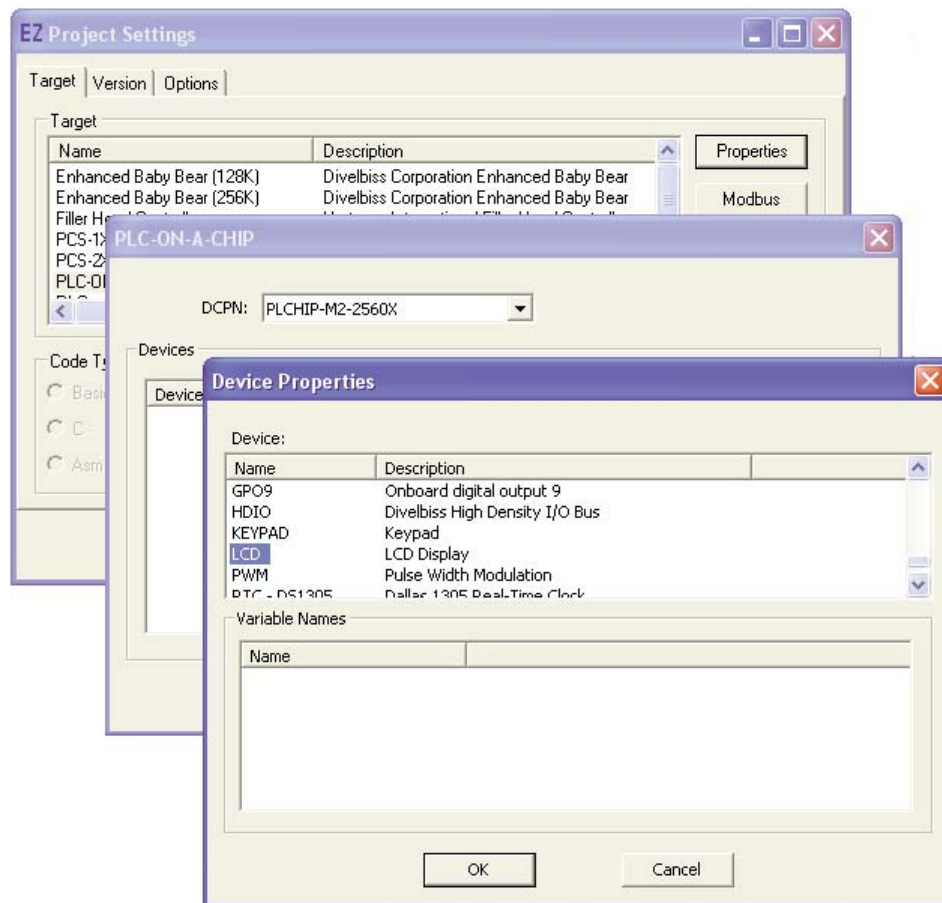


Figure 13.1

5. Click **OK** to install the LCD Display and return to the target's properties dialog.
6. Click on **LCD** and a **Properties** button will appear on the right side of the target properties dialog.
7. Click the **PROPERTIES** button. The LCD Properties dialog will open. This dialog is used to configure the LCD display on the target. See Figure 13.2.

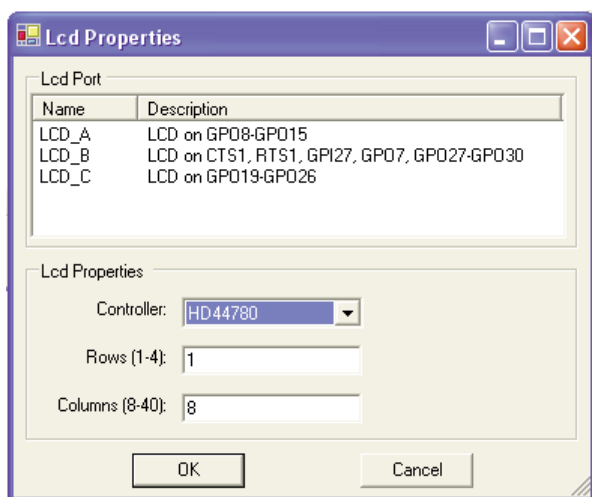


Figure 13.2

8. Select the **LCD PORT** to use from the available ports. You may use any of the listed ports provided it is not being used by another feature..
9. Select the **Controller**. Currently the HD44780 is the only supported controller.
10. Enter the number of **ROWS** on the display (1-4).
11. Enter the number of **COLUMNS** on the display (8-40).
12. Click **OK** to close the LCD Properties and save the changes.
13. Click **OK** to close the Target Settings Properties and save the changes.
14. Click **OK** to close the ProjectSettingsForm dialog and save the changes.

The LCD display is now ready for use once the display is physically connected. Refer to Section 20 - EZ LADDER Functions for using the LCD\_PRINT & LCD\_CLEAR functions.

# SECTION 14

## MATRIX KEYPAD



## Keypad Functionality

The Keypad feature provides that ability to add a 4x5 keypad matrix and monitor it from the ladder diagram. The Keypad functionality must be 'installed' on the target before it may be used. For information on connections and compatibility of the keypad, please refer to the PLC on a Chip Circuit Design Guidelines document.

To install the Keypad in the target configuration:

1. Use the menu, select *PROJECT....SETTINGS*. The ProjectSettingsForm dialog will open. Select the target.
2. Click the **PROPERTIES** button.
3. Click the **ADD** button.
4. From the provided ADD dialog *Devices* list, select the Keypad. See Figure 14.1.

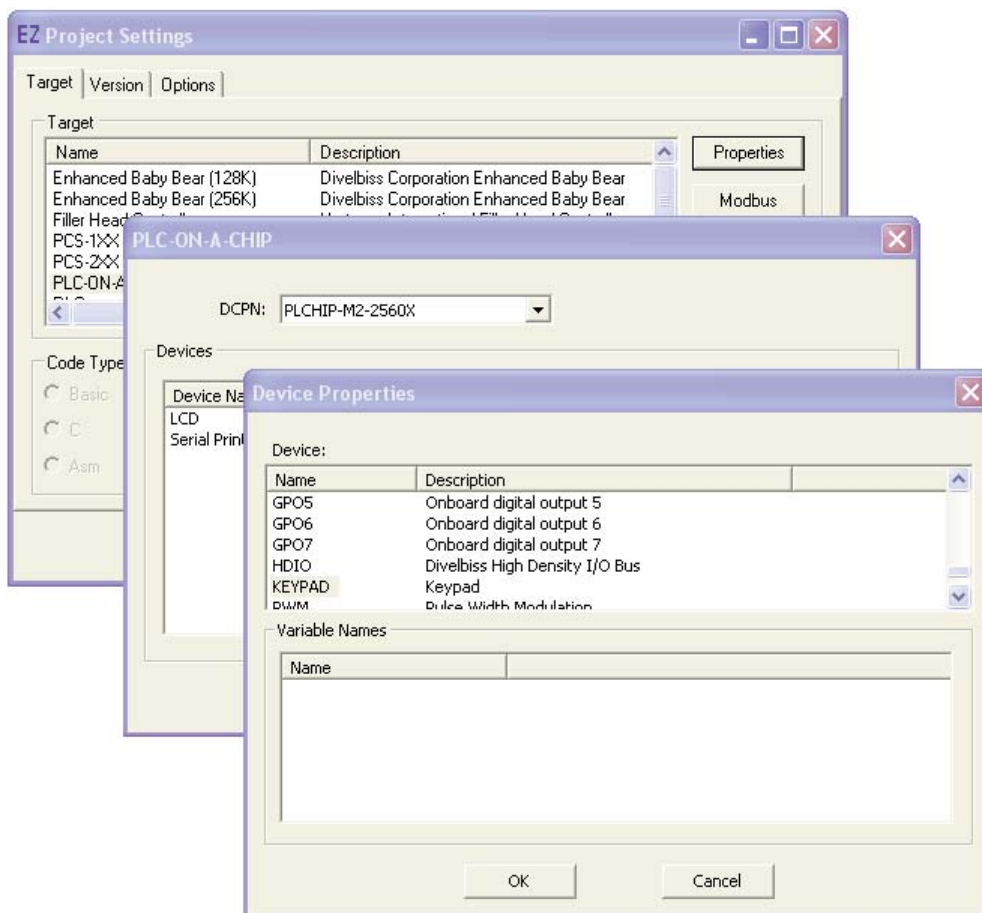


Figure 14.1

5. Click **OK** to install the Keypad and return to the target's properties dialog.
6. Click on Keypad and a Properties button will appear on the right side of the target properties dialog.
7. Click the **PROPERTIES** button. The Keypad Properties dialog will open. This dialog is used to configure the Keypad on the target. See Figure 14.2.

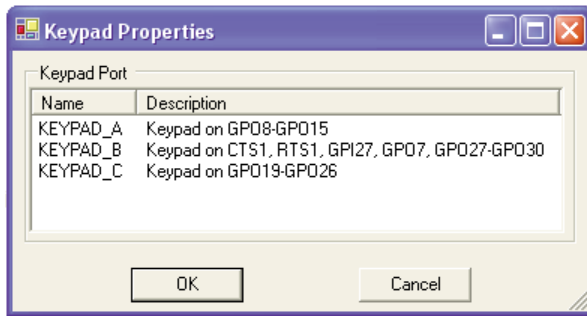


Figure 14.2

8. Select the **KEYPAD PORT** to use from the available ports. You may use any of the listed ports provided it is not being used by another feature..
9. Click **OK** to close the Keypad Properties and save the changes.
10. Click **OK** to close the Target Settings Properties and save the changes.
11. Click **OK** to close the ProjectSettingsForm dialog and save the changes.

The Keypad is now ready for use once the keypad is physically connected. Refer to Section 20 - EZ LADDER Functions for using the KEYPAD function. The Keypad is read using the Keypad function but individual keys may be read as a discreet digital input.

# SECTION 15

## J1939 COMMUNICATIONS



## J1939 Communications

CAN network enabled PLC on a Chip products support monitoring common J1939 broadcast parameters. J1939 must be enabled (installed) in the target settings (project settings) and is only available on targets with CAN network capability.

### J1939 Target Configuration

Before J1939 parameters may be received, the J1939 communications must be installed in the target settings.

Divelbiss standard controllers based on PLC on a Chip (Enhanced Baby Bear, PCS-XXX, etc) are configured based on the part number. To configure, select the correct part number. Click **J1939 PROPERTIES** and jump to step 8 to configure the serial port.

#### To install J1939 in the target configuration for PLC on a Chip:

1. Use the menu, select *PROJECT....SETTINGS*. The ProjectSettingsForm dialog will open. Select the target.
2. Click the **PROPERTIES** button.
3. Click the **ADD** button.
4. From the provided ADD dialog *Devices* list, select the J1939. See Figure 15.1.

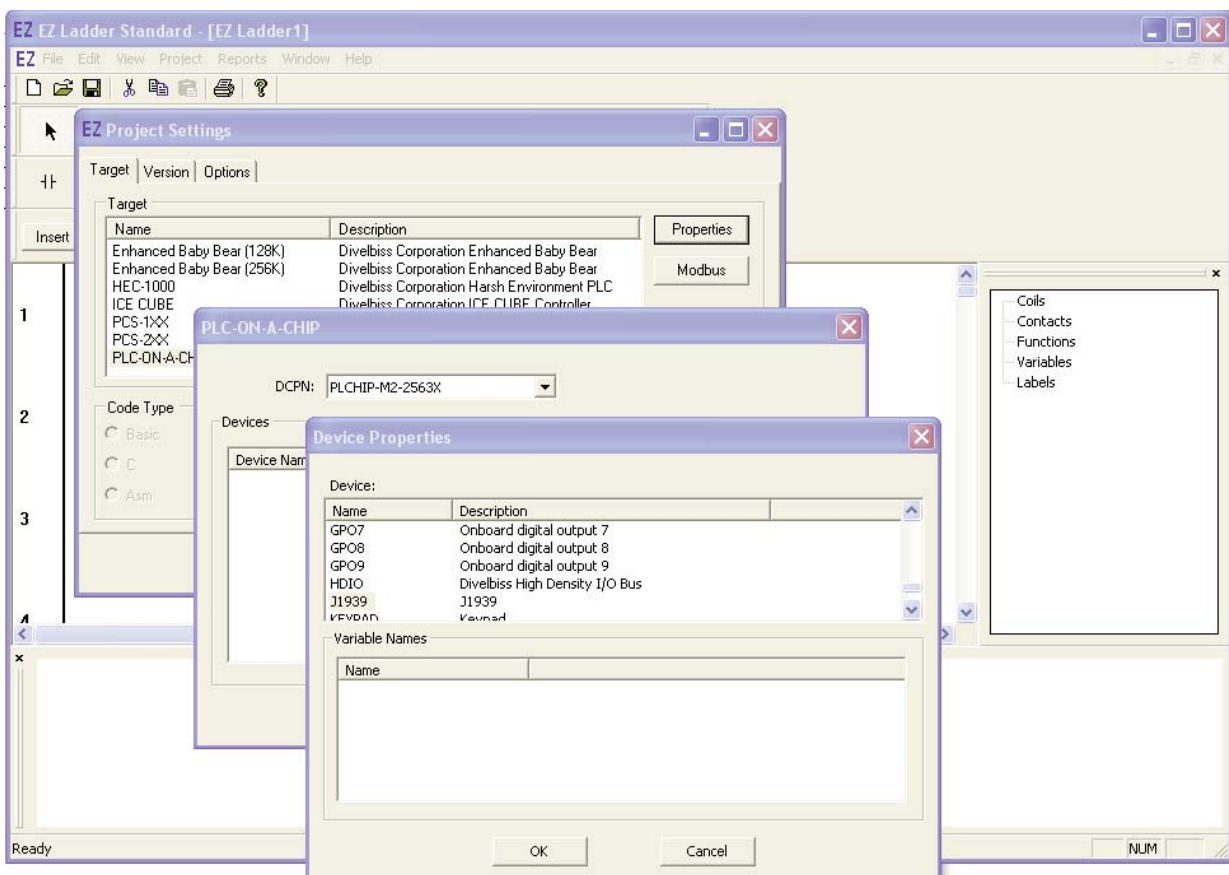


Figure 15.1

5. Click **OK** to install J1939 and return to the target's properties dialog.
6. Click on J1939 and a **Properties** button will appear on the right side of the target properties dialog.
7. Click the **PROPERTIES** button. The J1939 Properties dialog will open. This dialog is used to configure the J1939 Communications on the target. See Figure 15.2.

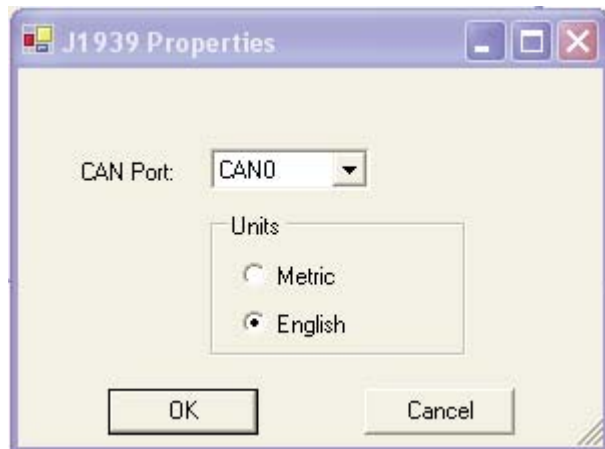


Figure 15.2

8. Select the **CAN Port** to use from the available using the drop-down menu. Note only one port per controller can use J1939.
9. Select the **Units**. Select Metric or English as the preferred unit of measure for engineering units..
10. Click **OK** to close the J1939Properties and save the changes.
16. Click **OK** to close the Target Settings Properties and save the changes.
17. Click **OK** to close the ProjectSettingsForm dialog and save the changes.

J1939 is now ready for use. Refer to Section 20 - EZ LADDER Functions for monitoring J1939 broadcasts using the J1939\_SPN Function and the supported Suspect Parameter Numbers (SPNs).



# SECTION 16

## OPTICAN NETWORK



## What is OptiCAN


OptiCAN is a Divelbiss proprietary CAN (Controller Area Network) that provides a communication link between Divelbiss OptiCAN enabled controllers and other OptiCAN enabled devices such as I/O modules and controllers. The Divelbiss OptiCAN network supports up to 64 nodes (devices) and is register based. Each node supports up to 256 registers and communication can be triggered based on time or on an event.


Divelbiss OptiCAN can perform the following major functions:


1. Allow controllers to access external I/O Devices
2. Allow controllers to access other controllers
3. Allow the user to configure devices utilizing the CAN protocol


Only Divelbiss OptiCAN enabled devices will communicate on the network.

## Planning your Network


 As with any network or communication scheme, the network should be planned taking into account the amount of communication, broadcast rate, communication triggers, register assignments and timing requirements. This plan is essential for a successful implementation of the network.

 All register needs should be identified and assigned for each device prior to the start of the programming. Register assignments should start at the high end of available registers and work backward (ie: start with register 127 and then assign 126 and so on). As some devices utilize lower register numbers this will ensure that the controller register assignments will not interfere with the device register assignments.

 While all controllers may 'broadcast' and 'listen', ideally one should be identified as the 'master' for the network. This master controller should be responsible for the network commands that start, stop and reset the OptiCAN network communications. These function blocks will be discussed later in this section.

 The last pages of the OptiCAN Network section are sample forms that may be used to aid in planning the network and assignments

## Hardware Requirements & Recommendations

 For optimal functionality, performance and noise immunity, all the hardware recommendations must be followed. A failure to follow recommended hardware requirements could result in decreased reliability of the OptiCAN Network.

Please adhere to the following requirements and recommendations:

1. The OptiCAN network cable should be of a 'twisted pair with shield' variety and cannot exceed 40 meters in total length. Additional length or incorrect cable type may limit functionality or cause the network to fail.

Please adhere to the following specifications for cable requirements for all OptiCAN networks.

**Twisted Pair Shielded Cable Specifications / Requirements**

Parameter	Symbol	Minimum	Nominal	Maximum	Unit	Comments
Impedance	Z	108	120	132	$\Omega$	
Specific Resistance	$r_b$	0	25	50	m $\Omega$ /m	
Specific Capacitance	$C_b$	0	40	75	pF/m	Between Conductors
	$C_s$	0	70	110	pF/m	Conductor to Shield

2. A 120 ohm resistor (load) is required at each end of the network. Please adhere to the following specifications for terminating resistor requirements for all OptiCAN networks.

**Terminating Resistor Specifications / Requirements**

Parameter	Symbol	Minimum	Nominal	Maximum	Unit	Conditions
Resistance	$R_L$	110	120	130	$\Omega$	minimum power dissipation 400 mW <sup>(1)</sup>
Inductance				1	$\mu$ h	

1. Assumes a short of 16V to  $V_{CAN\_H}$

3. The cable shield should be ideally grounded near the middle of the network (cable) run to earth ground. Please note, the shield should only be connected to ground and one point on the network. Multiple ground points could cause a ground loop, decrease noise immunity and adversely affect network performance.
4. If wiring as a network bus with stub connections, the maximum stub length from bus to node is 1 Meter. See Figure 16.0 for a sample connection diagram.

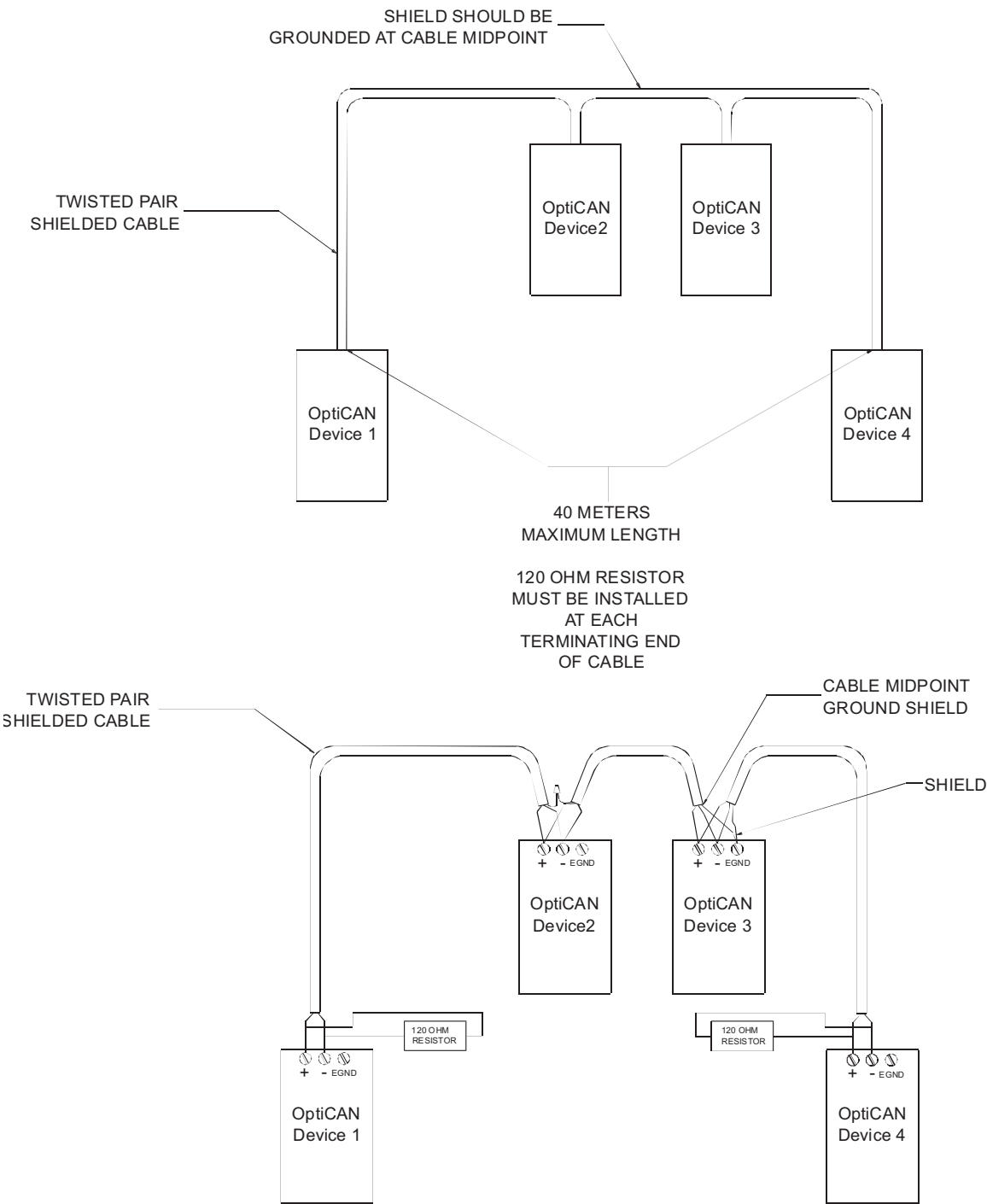


Figure 16.0

## OptiCAN Specifications

**Bandwidth:** OptiCAN network operates at 250KBits / Sec

**Maximum Cable Length:** The entire network's cable length should not exceed 40 meters.

**Maximum # of Nodes:** The maximum number of nodes (devices) that may be connected to one network is 64.

**Registers per Node:** Each node supports 256 registers.

## OptiCAN Controller Operation

A typical application involves a controller 'running' its own program, monitoring inputs and controlling outputs based upon the program that is running. When connected to an OptiCAN network the same holds true, but now the network functionality is gained.

The following describes how a controller operated when used on a active OptiCAN network.

A controller on an OptiCAN network will function as normal, monitoring its inputs and controlling outputs based on local logic.

The controller can 'broadcast' across the network for other devices (I/O and controllers) to receive and act upon. Broadcasting is the transmission of packets of data. These packets of data include general formatting, the contents of and identifying attributes of registers that are set in the ladder program.

In addition to broadcasting data, the controller can also 'listen' for broadcasts from other devices. These broadcasts are received in registers and then the controller can make decisions and act upon the results.

## OptiCAN Controller 'Heartbeat'



Each OptiCAN controller has the ability to broadcast a 'heartbeat'. This signal is broadcast at a regular interval. This is used to ensure that all devices on the network are communicating properly. Each node automatically monitors this heartbeat. In the event the heartbeat is 'lost', then the local ladder program should ignore data from the network as the loss of heartbeat signifies that communication with part or all of the network has been lost. By using the 'OPTICAN\_NODESTATUS' function block, the controller (or user) can determine if a node has network problems.

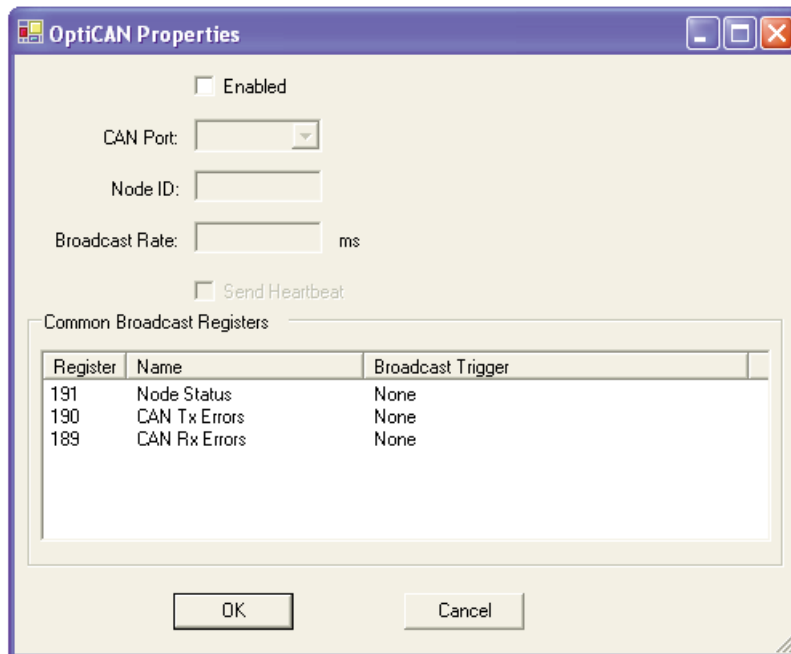


One node on the OptiCAN network MUST broadcast the heartbeat message for the network to function properly. Although it is possible to have multiple controllers on one network sending heartbeats, it is recommended only one controller send a heartbeat per network.

## Configuring a Controller on the OptiCAN Network

Before a controller can be used on the OptiCAN network, the controller must be configured using EZ LADDER. Using EZ LADDER, create or open the project (program) that will be installed (loaded) into the controller. To configure a controller:

1. From the main screen (Edit Mode), using the top menu's, click *Project....Settings*. This will open the Project Settings dialog box.
2. Select the target of the controller and then click the **PROPERTIES** button. The Target Properties dialog box will open.
3. Click the **OPTICAN PROPERTIES** button to open the OptiCAN Properties dialog box. See Figure 16.1.



**Figure 16.1**

4. Click on the *Enabled* checkbox to 'check it'. This activates the OptiCAN for the target. When selected, the other previously not available boxes become available. See Figure 16.2.
5. Using the *CAN Port* drop down box, select the correct CAN port the network will be using. Please note, different CAN Port options will be shown dependent upon the actual target selected. See Figure 16.2.
6. In the *Node ID* text box, type in the node ID of the controller (network address). See Figure 16.2.
7. In the *Broadcast Rate* text box, type in the broadcast rate for the controller in milliseconds. This timing should have been determined during the network planning phase. See Figure 16.2.
8. Click on the *Send Heartbeat* checkbox if required. When checked, the controller will send a 'heartbeat' across the network. This 'heartbeat' may be used to determine if communication is functional to other devices. See Figure 16.2.

9. Common broadcast registers may be configured here. These registers can be configured to 'broadcast' at a specified interval, On a Change of State or both. If required for your application, click on the register and use the drop down box to select a Broadcast trigger. See Figure 16.2.

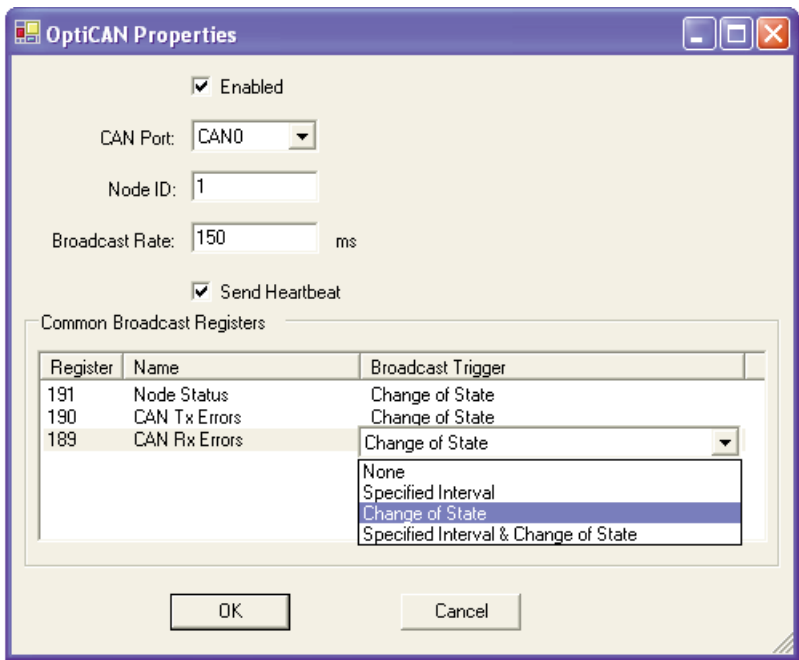


Figure 16.2

Broadcasting / Sending Data from a Controller to another Device

Broadcasting or sending data to a network node is configured using variables. To broadcast, click the Insert Variable button and click **ADD** to add a new variable or click the Edit Variable button, select a variable and click **EDIT** to edit the variable. The Add Variable or Edit Variable dialog box will open. See Figure 16.3.

Inst  
Vars

Edit  
Vars

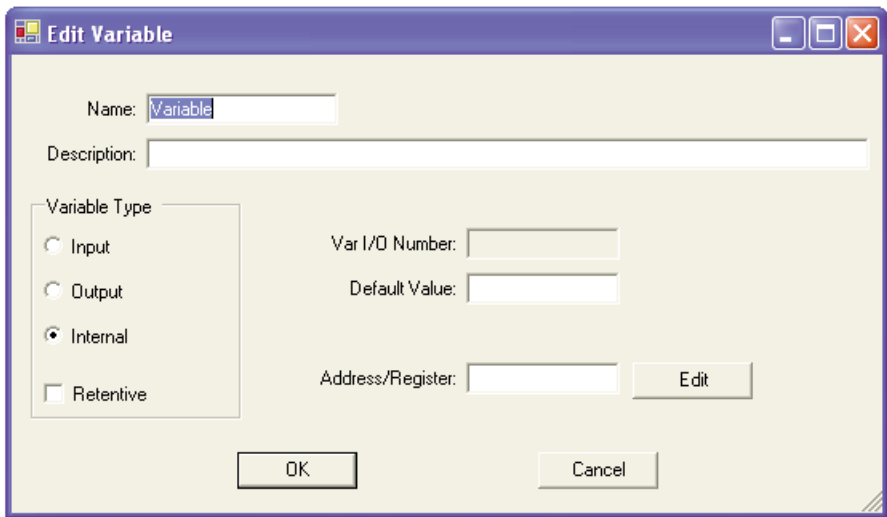


Figure 16.3

As any other variable, the variable **NAME** must be entered (if a new variable is being added). The **DESCRIPTION** is an optional field to give a description of the variable for reference only. The **VARIABLE TYPE** must be set as with any other variable. If the variable is to be retentive, the **RETENTIVE** checkbox must be checked. If the variable type is an input or output, then the **VAR I/O NUMBER** must be set. The **DEFAULT VALUE** may optionally be set if the variable type is internal. See manual [Section 5 - Creating Ladder Diagrams](#) for more details on adding or editing variables and variable types.

The last box to configure is the **ADDRESS / REGISTER** text box. This box is used to identify special functions of the variable like register number or address. This box is used to configure the variable to broadcast over the OptiCAN network. See Figure 16.4.

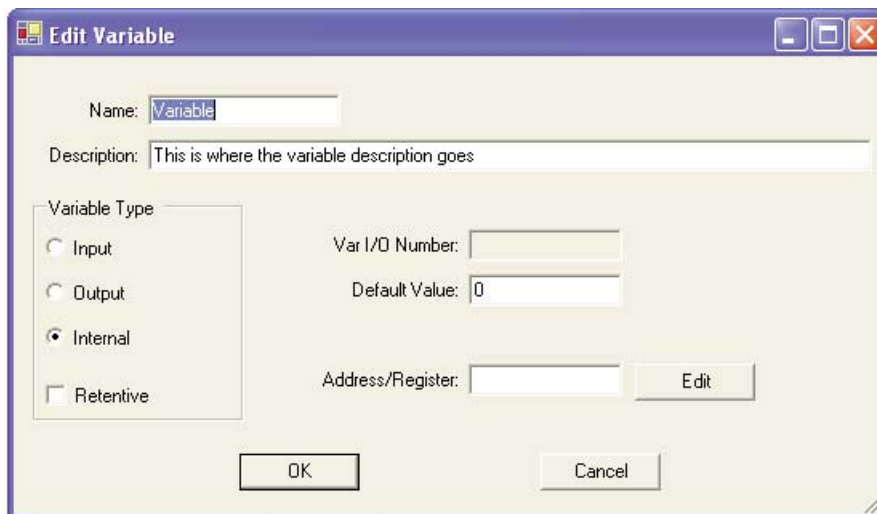


Figure 16.4

Click the **EDIT** button next to the Address/Register box. The *Edit Address / Register* dialog box will open. Click on the drop down menu and select **CAN\_ (OptiCAN)** from the choices. This enables this variable to access the OptiCAN network. See Figure 16.5.

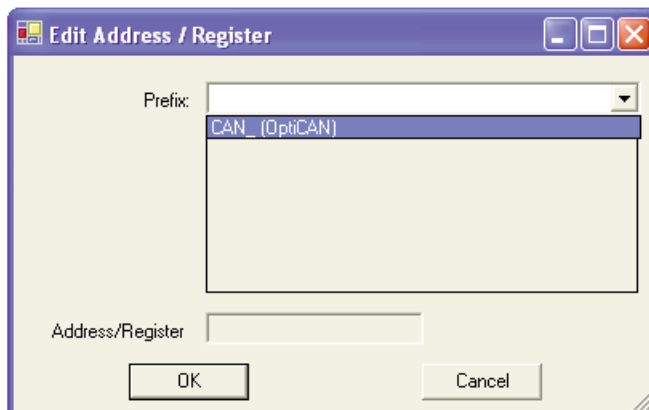


Figure 16.5

When selected, other OptiCAN entry boxes will appear and be editable.



Enter the node identification number of the device you are broadcasting to in the **Node ID** box. If left blank, the broadcast will be global (to all nodes). Enter the register number on the remote device the broadcast should be sent to in the **Register Number** box. From the **Broadcast Trigger** drop down menu, select the broadcast trigger for this variable (None, Specified Interval, Change of State or Specified Interval and Change of State). Do not use the **IN** checkbox at this time. See Figure 16.6.

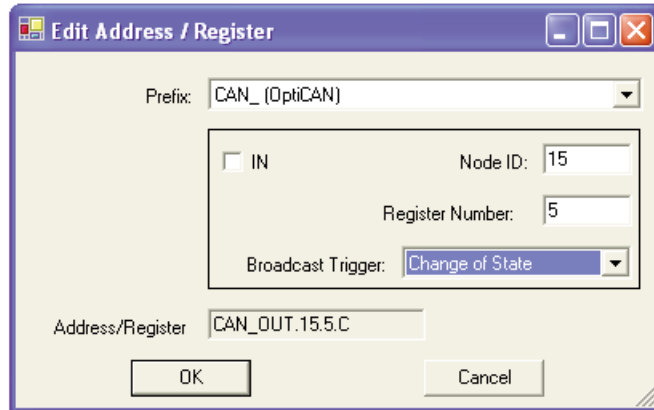


Figure 16.6

Click **OK** to save the variable information.

This variable will now broadcast as configured over the OptiCAN network.

## Receiving Data on a Controller from another Device

To retrieve data from a network node is configured using variables. To receive, click the **Inst Vars** Insert Variable button and click **ADD** to add a new variable or click the Edit Variable button, select a variable and click **EDIT** to edit the variable. The Add Variable or Edit Variable dialog box will open. See Figure 16.3.

As any other variable, the variable **NAME** must be entered (if a new variable is being added). The **DESCRIPTION** is an optional field to give a description of the variable for reference only. The **VARIABLE TYPE** must be set as with any other variable. If the variable it to be retentive, the **RETENTIVE** checkbox must be checked. If the variable type is an input or output, then the **VAR I/O NUMBER** must be set. The **DEFAULT VALUE** may optionally be set if the variable type is internal. See manual [Section 5 - Creating Ladder Diagrams](#) for more details on adding or editing variables and variable types.

The last box to configure is the **ADDRESS / REGISTER** text box. This box is used to identify special functions of the variable like register number or address. This box is used to configure the variable to receive data from the OptiCAN network. See Figure 16.4.

Click the **EDIT** button next to the Address/Register box. The *Edit Address / Register* dialog box will open. Click on the drop down menu and select *CAN\_ (OptiCAN)* from the choices. This enables this variable to access the OptiCAN network. See Figure 16.5.

When selected, other OptiCAN entry boxes will appear and be editable.

Enter the node identification number of the device you wishing to 'listen for' in the **Node ID** box (If left blank, any broadcast with the same register number will be received). Enter the register number in the **Register Number** box that it will 'listen for' in the broadcast packet. Click the **IN** checkbox to identify this variable as an input. Note, the **Broadcast Trigger** drop down menu is disabled. See Figure 16.7.

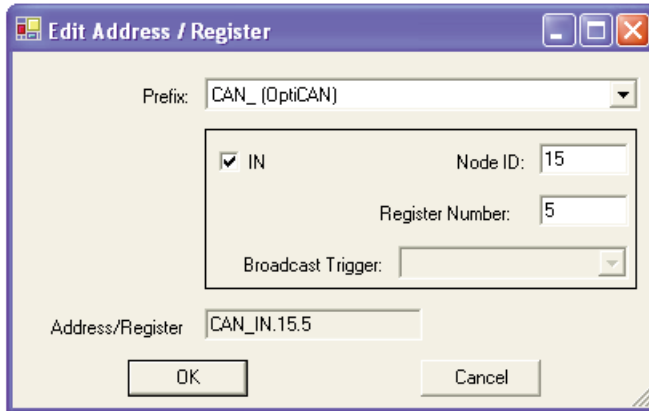


Figure 16.7

Click **OK** to save the variable information.

This variable will 'listen' and update based on remote broadcasts from the configured Node ID and Register over the OptiCAN network.

### The OPTICAN\_NODESTATUS Function Block

This function block is used to monitor the status of any Node on the network (provided the node has been configured to broadcast its *Status*).

The function is **OPTICAN\_NODESTATUS**. It can be placed from the functions drop down list (if OptiCAN enabled and configured properly).

For this function to operate properly, the remote node you will be 'listening' for must be configured to broadcast its node status in a time interval less than the timeout value for the function.

To use the **OPTICAN\_NODESTATUS** function, select the function from the *Insert Function* drop down list. Locate in the ladder program and click where the function is to be placed. The *OptiCAN Node Status Properties* dialog box will open. See Figure 16.8.

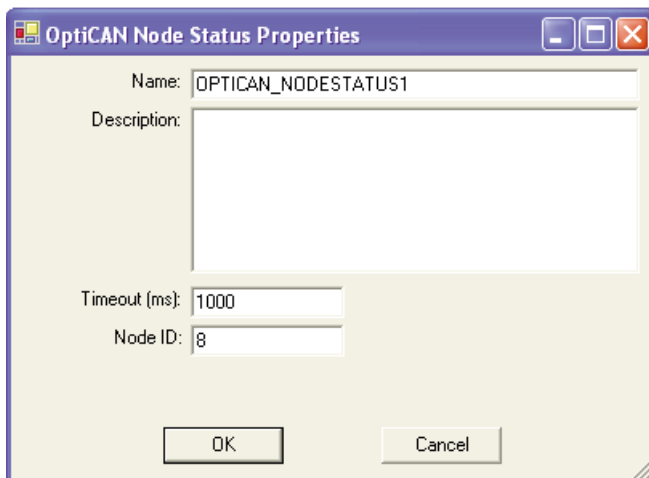


Figure 16.8

The function **NAME** can be changed if preferred or the default name may be used. The **DESCRIPTION** is an optional field to enter a description of what the function does for later reference. The TIMEOUT (ms) box is where a timeout value in milliseconds is entered. The NODE ID is the node ID number of the remote device that will be listened to for status. Click **OK** to place the function. See Figure 16.9.

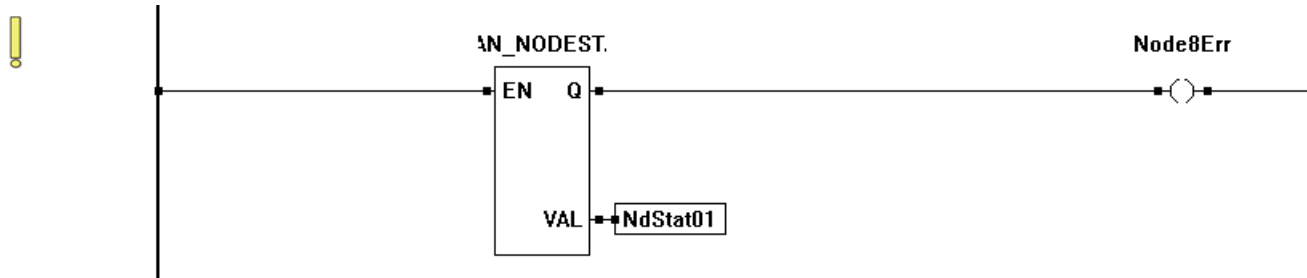


Figure 16.9

**EN Input:** This input enables the function to operate. A true input will enable the function.

**Q Output:** If the node status is not received during Timeout (ms) period, the function will timeout indicating an error has occurred (the normally high Q output will go low on error).

**VAL Output:** This is a 32-bit integer number. The upper 16-bits stores the error code while the lower 16-bits stores the node status. A list of error codes can be found at the end of the this OptiCAN network section.

 To monitor the status of 'your own node', in the Add/Edit variable dialog, check the 'IN' box, leave the Node ID blank and enter register # 191.

### The OPTICAN\_TXNETMSG Function Block

This function block is used to send start, stop and reset messages to all nodes on the OptiCAN network. On power-up, the OptiCAN network does not start as a default. The OptiCAN network communication must receive a start command from a controller. To use the start, stop or reset messages, the **OPTICAN\_TXNETMSG** function block is used.

To use the **OPTICAN\_TXNETMSG** function, select the function from the *Insert Function* drop down list. Locate in the ladder program and click where the function is to be placed. (This function is only available on OptiCAN supported targets and if the OptiCAN network has been enabled). The *OptiCAN Transmit Network Message* dialog box will open. See Figure 16.10.

The function **NAME** can be changed if preferred or the default name may be used. The **DESCRIPTION** is an optional field to enter a description of what the function does for later reference. The TIMEOUT (ms) box is where a timeout value in milliseconds is entered. The **NETWORK MESSAGE** drop down box selects the type of message to globally transmit to all nodes.

The 'Start Network' command that must be sent to start the OptiCAN network communications

The 'Stop Network' command halts the OptiCAN network communications.

The 'Reset Network' command is used to cause a hard reset on the OptiCAN network.

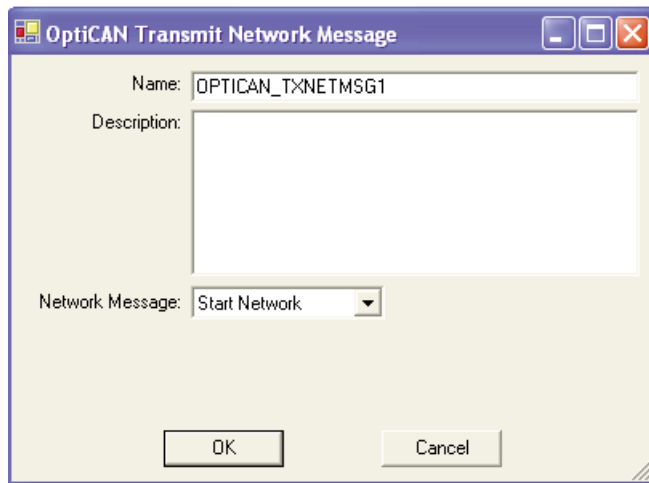



Figure 16.10

Click **OK** to place the function in the ladder program. See Figure 16.11.

 Typically, these commands should only be sent when required. For example, the 'Start Network' command should be sent on power-up and then only when specifically required after that (in case of an error). This function block will only send the command on the rising edge of EN. If multiple broadcasts are required, additional logic will be required to 'repeat' the broadcast.

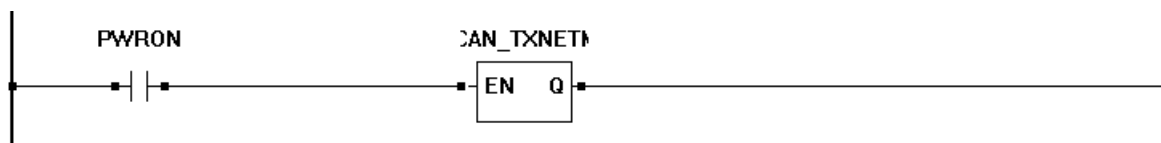



Figure 16.11

## Controller Node Registers

OptiCAN controller network registers are numbered 0-255. Registers 0 to 127 are not assigned and are to be used in the ladder program to communicate over the OptiCAN network. Registers 128 to 255 are pre-assigned for other functions. See the list of register assignments later in this OptiCAN network section.

 It is recommended that before programming is started that all nodes are identified, assigned a node ID and documented. For each device, their register requirements should be identified, registers assigned and registers documented by the programmer. This will verify the all requirements are met and help to promote proper functionality and design.

## Configuring Other OptiCAN Devices (Non-Controller)

Configuring other OptiCAN devices, such as I/O must be done using the built-in **Divebiss OptiCAN Configuration Tool** or the optionally purchased **OptiCAN Configuration Tool Pro**.

### OptiCAN Configuration Tool Professional

The OptiCAN Configuration Tool Pro should be used when a number of nodes exceeds 10 or if you want to see more network data than the built-in OptiCAN Configuration Tool can provide. The Professional (Pro) version allows for more nodes, viewing real time status, sending network Start, Stop and Reset Commands and additional network data for troubleshooting. The Professional version connects with a Divebiss supplied USB to CAN hardware 'dongle' independent of any controllers. Refer to the OptiCAN Configuration Tool Pro Manual for more information about the OptiCAN Configuration Tool Pro.

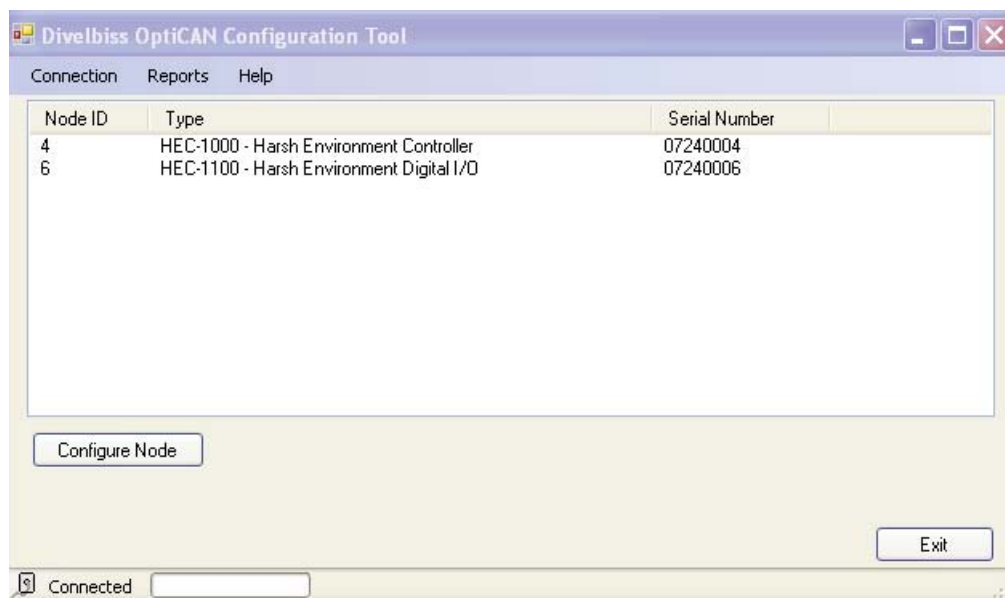
### Built-in OptiCAN Configuration Tool

The built-in OptiCAN Configuration Tool can configure and detect up to 10 nodes giving Node ID, Type and Serial Number.

The OptiCAN Configuration Tool allows the configuration of up to 10 non-controller nodes, gives an overview of all devices connected to the OptiCAN network. To communicate, the OptiCAN Configuration Tool uses the EZ LADDER target communication configuration.

To open the OptiCAN Configuration Tool, your programming **MON** must be connected to a controller, have EZ LADDER open and have a ladder project loaded. Click the **MON** Monitor button on the toolbar to enter EZ LADDER's Monitor Mode.

From the Monitor Mode, one the menus, click *Project....OptiCAN*. This will open the Divebiss OptiCAN Configuration Tool in a new window. See Figure 16.12.



**Figure 16.12**

Please note, EZ LADDER must have a project loaded and be in Monitor mode (with OptiCAN enabled) to open the OptiCAN Configuration Tool. It is not necessary to connect to the target controller. If connected to the controller, the OptiCAN Configuration Tool will disconnect EZ LADDER from the controller when it opens. When the OptiCAN Configuration tool connects, it will send the Stop Command for the network automatically. The network will have to be restarted for proper operation.

As shown in Figure 16.12, there are two devices on the connected OptiCAN network. The tool shows the NODE ID, Type and Serial Number. These two devices have already been configured as they have Node ID's assigned.



When configuring a non-controller device for the first time, the device will display with a Node ID of 255. The '255' designation is reserved for devices that have not been configured. See Figure 16.13. For multiple new devices, they will all be assigned the same '255' Node ID. The controller can differentiate between devices that have not been configured using the Serial Number. The serial number is programmed at the factory and cannot change.



Please note, only non-controller device Node IDs may be changed using this tool. Controller Node IDs are only changeable in the actual target nodes ladder program.

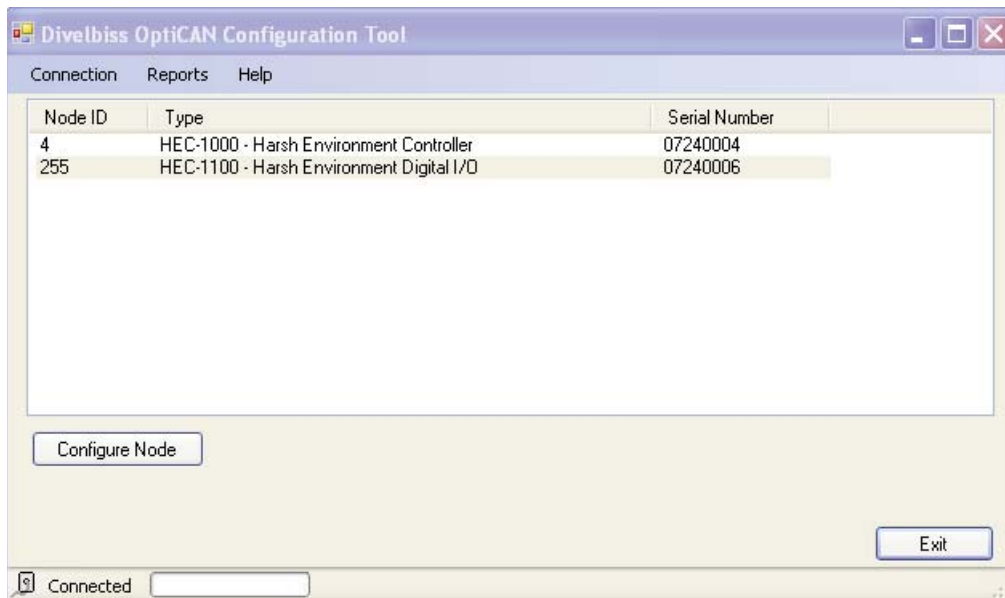


Figure 16.13

To configure a node, highlight the node in the list and click **CONFIGURE NODE**. A *Node Configuration* dialog box will open. See Figure 16.14.

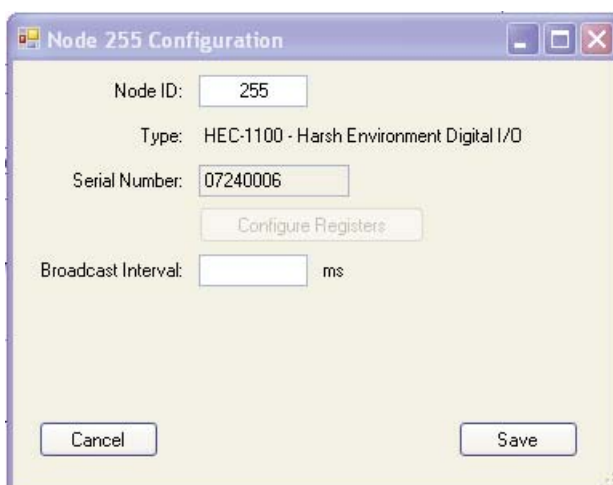


Figure 16.14

The following can be viewed from the *Node Configuration* dialog. Some items are configurable.

<b>Node ID:</b>	This is where the node ID number is set.
<b>Type:</b>	This is the description of the device (not editable).
<b>Serial Number:</b>	This is the serial number of the devices (programmed at factory and is not editable).
<b>Broadcast Interval:</b>	This is the interval (rate) at which the registers will be broadcast on the network.

The **CONFIGURE REGISTERS** button is used to configure the registers of the device including the trigger and value. Click the **CONFIGURE REGISTERS** button to open the *Configure Registers* dialog box. See Figure 16.15.

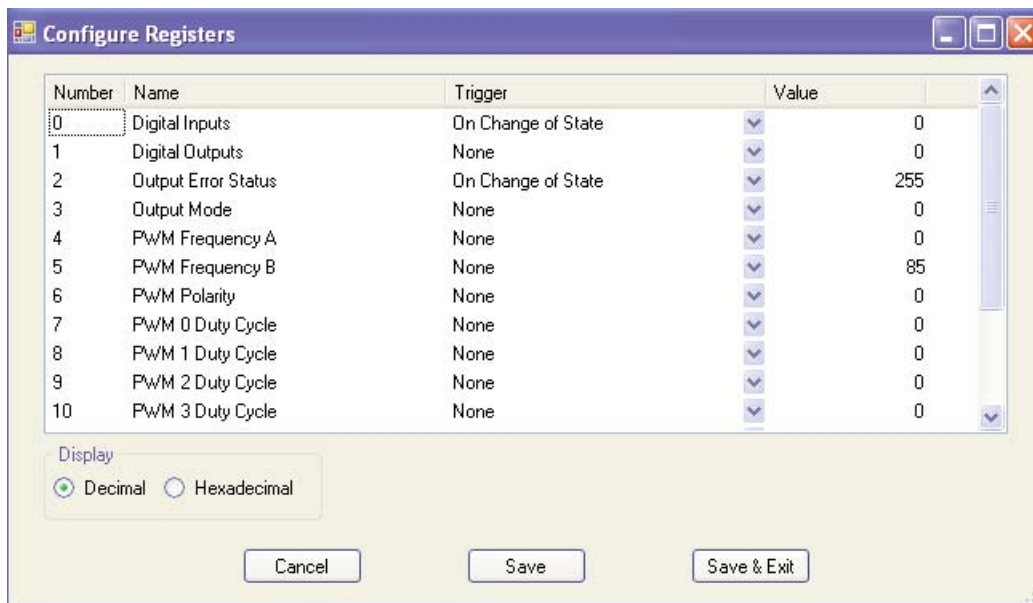


Figure 16.15

To change the Trigger for any register, highlight the register and click the down arrow of the trigger column for the register that requires changing. This will open a small list of available trigger options. Each register maintains its own individual trigger setting. See Figure 16.16.

When each of the registers of the node have been configured, click the **SAVE & EXIT** button to save changes and close the *Configure Registers* dialog box and return to the *Node Configuration*.

In addition to changing the trigger, from this dialog, the Value for each register can be changed (providing the register is writable). The numbers can be represented in decimal or in hex. Figure 16.16 is set to display in decimal. As an example, register number 1 (Digital Outputs) will directly control the outputs on the node. By changing the Value, the outputs can be made to be on or off.

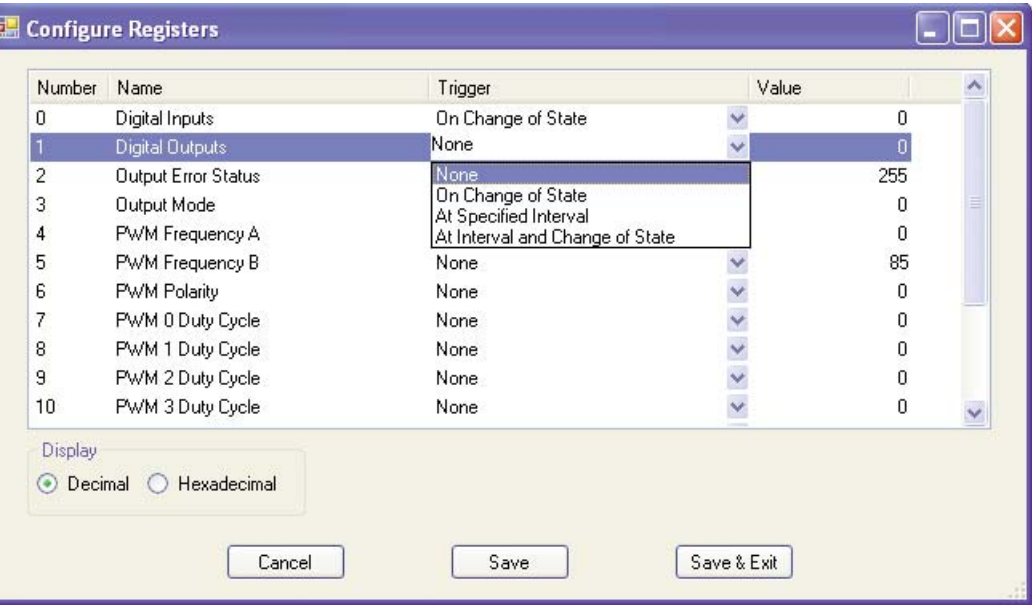


Figure 16.16

The numbers that are to be entered is a decimal representation of binary bits that correspond to the outputs themselves.

Decimal Number	128	64	32	16	8	4	2	1	0
Corresponding Output	8	7	6	5	4	3	2	1	All Off

Setting the value of the output register to 128 will cause only output 8 on the node to be ON.

Setting the value of the output register to 8 will cause only output 4 of the node to be ON.

If the value of the output register is set to 40, output 6 and output 4 will both be ON ( output 4 bit = Decimal 8, output 6 bit = Decimal 32. 32 + 8 = Decimal 40).

The number shown as examples above are for decimal only. If using Hexidecimal, they must be converted appropriately.

Changes here are immediate after clicking **SAVE** or **SAVE& EXIT**.



## OptiCAN Network Register Assignments

The OptiCAN network operates based on preset and user defined registers. The following are general register assignments and information common for all OptiCAN enabled controllers and devices. For non-controller devices, please consult the product's datasheet for detailed register assignments and preset functions.

### General Register Assignments

These are the overall general register assignments common to all OptiCAN enabled devices.

Register Number	Assigned Use
0 to 127	User Defined Registers (Controller), Device Defined Registers (I/O & Other Devices)
128 to 191	Common Broadcast Registers
192 to 255	Common Configuration & Command Registers

User Defined registers for controllers are available for the user to define the use of during the ladder diagram development. Device Defined registers for I/O and other devices have preset definitions of register use and cannot be changed.

### Common Configuration / Command Registers

These registers are pre-assigned and cannot be altered.

Register Number	Name	Description	Read / Write
255	Node ID	This Node's ID Number	Read
254	Serial Number	This Node's Serial Number	Read
253	Broadcast Interval	Interval for Broadcasting (ms)	Read / Write
252	Broadcast Trigger 0	Broadcast Trigger for Registers 0 to 15	Read / Write
251	Broadcast Trigger 1	Broadcast Trigger for Registers 16 to 31	Read / Write
250	Broadcast Trigger 2	Broadcast Trigger for Registers 32 to 47	Read / Write
249	Broadcast Trigger 3	Broadcast Trigger for Registers 48 to 63	Read / Write
248	Broadcast Trigger 4	Broadcast Trigger for Registers 64 to 79	Read / Write
247	Broadcast Trigger 5	Broadcast Trigger for Registers 80 to 95	Read / Write
246	Broadcast Trigger 6	Broadcast Trigger for Registers 96 to 111	Read / Write
245	Broadcast Trigger 7	Broadcast Trigger for Registers 112 to 127	Read / Write
244	Broadcast Trigger 8	Broadcast Trigger for Registers 128 to 143	Read / Write
243	Broadcast Trigger 9	Broadcast Trigger for Registers 144 to 159	Read / Write
242	Broadcast Trigger 10	Broadcast Trigger for Registers 160 to 175	Read / Write
241	Broadcast Trigger 11	Broadcast Trigger for Registers 176 to 191	Read / Write

### Common Broadcast Registers

These registers are pre-assigned and cannot be altered.

Register Number	Name	Description	Read / Write
191	Node Status	This Node's Status	Read
190	CAN TX Errors	CAN Transmit Error Counter	Read
189	CAN RX Errors	CAN Receive Error Counter	Read

Register 191 (Node Status) is a 32-bit number. The lower 16-bits is the Status Code (Reset, Active, other). The upper 16-bits is the Error Code. The Error codes are split into two groups:

0 to 32767	Device Specific Errors
32768 to 65535	Common Error Codes.

#### Status Codes

1 = Reset  
2 = Active  
4 = Reset

#### Common Error Codes:

65535 = CAN Controller Receive Error  
65534 = CAN Controller Receive Warning  
65533 = CAN Controller Transmit Error  
65532 = CAN Controller Transmit Warning  
65531 = CAN Controller Bus Off State  
65530 = CAN Controller Data Overrun  
65519 = OptiCAN Heartbeat Timeout  
65518 = CAN Controller Error

### OptiCAN Node List Notes

Notes can be added to the list of nodes to help with documentation and service later. This is accessed from the OptiCAN Configuration Tool. To access this feature, click *Reports....Node List*. The Node List Report window will open. Place the cursor under the Note Heading next to the node of choice. Simply type in the notes for that node.

The node list and notes may be saved and printed for future reference.

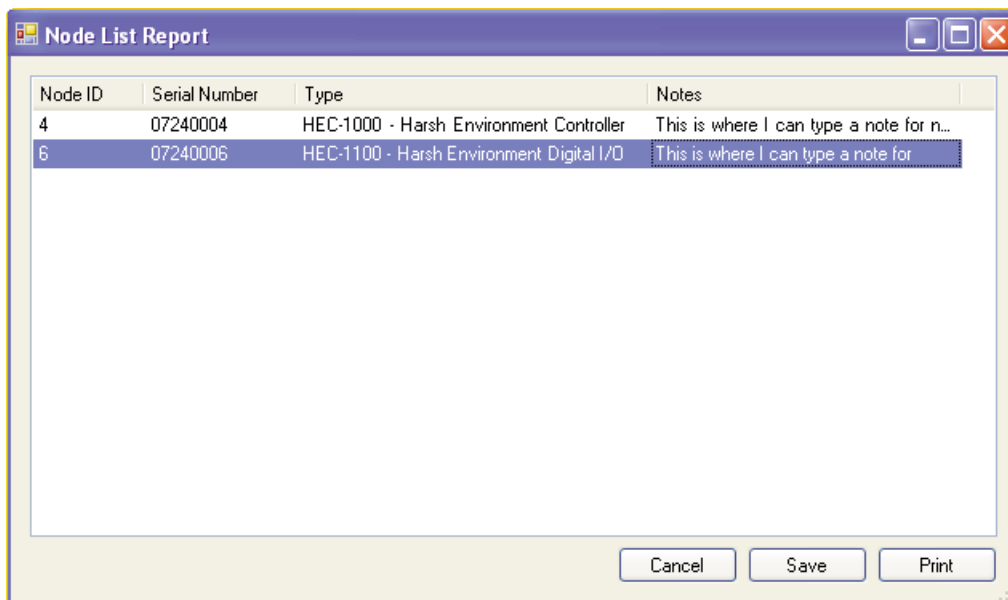


Figure 16.17

# OptiCAN Controller Network Planning

This form is to be used with OptiCAN enabled controllers only for the purpose of identifying, planning and logging network registers and ID's

Controller Network Setup

Controller Name:

Node ID:

Serial #:

Controller Description:

Broadcast Heartbeat?

Y

N

Will this controller Broadcast Start/Stop/Reset?

Y

N

Broadcast Rate:

ms

Register	Name	Broadcast Trigger			
191	Node Status	<input type="checkbox"/> None	<input type="checkbox"/> Specified Interval	<input type="checkbox"/> Change of State	<input type="checkbox"/> Specified Interval & Change of State
190	CAN Tx Errors	<input type="checkbox"/> None	<input type="checkbox"/> Specified Interval	<input type="checkbox"/> Change of State	<input type="checkbox"/> Specified Interval & Change of State
189	CAN Rx Erros	<input type="checkbox"/> None	<input type="checkbox"/> Specified Interval	<input type="checkbox"/> Change of State	<input type="checkbox"/> Specified Interval & Change of State

Input Register Assignments (Receive Broadcast)			
Register # to Listen for	Description	Variable Assignment	Node ID to Listen For
0			
1			
2			
3			
4			
5			
6			
7			
8			
9			
10			
11			
12			
13			
14			
15			
16			
17			
18			
19			
20			
21			
22			
23			
24			
25			
26			
27			
28			
29			
30			
31			
32			
33			
34			
35			

# Input Register Assignments (Receive Broadcast)

Register # to Listen for	Description	Variable Assignment	Node ID to Listen For
36			
37			
38			
39			
40			
41			
42			
43			
44			
45			
46			
47			
48			
49			
50			
51			
52			
53			
54			
55			
56			
57			
58			
59			
60			
61			
62			
63			
64			
65			
66			
67			
68			
69			
70			
71			
72			
73			
74			
75			
76			
77			
78			
79			
80			
81			
82			
83			
84			
85			
86			
87			
88			
89			
90			
91			
92			
93			

### Input Register Assignments (Receive Broadcast)

Register # to Listen for	Description	Variable Assignment	Node ID to Listen For
94			
95			
96			
97			
98			
99			
100			
101			
102			
103			
104			
105			
106			
107			
108			
109			
110			
111			
112			
113			
114			
115			
116			
117			
118			
119			
120			
121			
122			
123			
124			
125			
126			
127			

Receive Register Notes:

### Broadcast Assignments (Transmit)

[illegible]

# SECTION 17

## TARGET FEATURES & FUNCTIONS LISTS



EZ LADDER is designed to interface and program multiple products (targets). Each target's features and supported functions may differ based on the design of the target and it's intended uses. This section describes each target's supported features and functions.

### PLC ON A CHIP (PLCHIP-M2-1280X)

The supported features include all features that are supported individually. Using certain features may limit availability of other features.

#### SUPPORTED FEATURES:

Analog Inputs ( 8 Channels)  
Real Time Clock (External)

Hardware Counter  
HDIO Bus

Retentive Variables

#### SUPPORTED FUNCTIONS:

Less Than (<)  
Less Than Equal To (<=)  
Not Equal To (<>)  
Equal To (=)  
Greater Than (>)  
Greater Than Equal To (>=)  
Absolute Value (ABS)  
Addition (ADD)  
Bitwise AND (AND)  
Average (AVG)  
Bit Pack (BIT\_PACK)  
Bit Unpack (BIT\_UNPACK)  
Convert to Boolean (BOOLEAN)  
Compare (CMP)  
Hardware Counter (CNTRTMR)  
Count Down (CTD)  
Count Up (CTU)  
Count Up / Down (CTUD)  
Division (DIV)  
Drum Sequencer (DRUM\_SEQ)  
Falling Edge Detect (F\_TRIG)  
Get Date (GETDATE)  
Get Time (GETTIME)  
High Speed Timer (HIGH\_SPD\_TMR)  
Hysteresis (HYSTER)  
Convert to Integer (INTEGER)  
Latching Coil (LATCH)

Limit (LIMIT)  
Moving Average (MAVG)  
Maximum (MAX)  
Minimum (MIN)  
Modulo (MOD)  
Multiplication (MULT)  
Bitwise NOT (NOT)  
Bitwise OR (OR)  
Rising Edge Detect (R\_TRIG)  
Convert to Real (REAL)  
Rotate Left (ROL)  
Rotate Right (ROR)  
Reset / Set -Reset Dominant (RS)  
Select (SEL)  
Set Date (SETDATE)  
Set Time (SETTIME)  
Shift Left (SHL)  
Shift Right (SHR)  
Set / Reset -Set Dominant (SR)  
Subtraction (SUB)  
Convert to Timer (TIMER)  
Time Delay Off (TOF)  
Time Delay On (TON)  
Pulse Timer (TP)  
Unlatching Coil (UNLATCH)  
Bitwise XOR (XOR)



### PLC ON A CHIP (PLCHIP-M2-2560X)

The supported features include all features that are supported individually. Using certain features may limit availability of other features.

#### SUPPORTED FEATURES:

Analog Inputs ( 8 Channels)	Hardware Counter	Retentive Variables
Real Time Clock (External)	HDIO Bus	Keypad Support
LCD Display Support	PWM Outputs	SPI Slave
Synchronous Serial Interface (SSI)	Serial Printing	Modbus Slave
EEPROM Storage		

#### SUPPORTED FUNCTIONS:

Less Than (<)	LCD Clear (LCD_CLEAR)
Less Than Equal To (<=)	LCD Print (LCD_PRINT)
Not Equal To (<>)	Limit (LIMIT)
Equal To (=)	Moving Average (MAVG)
EEPROM Read (EEPROM_READ)	Maximum (MAX)
EEPROM Write (EEPROM_WRITE)	Minimum (MIN)
Greater Than (>)	Modulo (MOD)
Greater Than Equal To (>=)	Multiplication (MULT)
Grey Scale Encoder (GC_SSI)	Bitwise NOT (NOT)
Absolute Value (ABS)	Bitwise OR (OR)
Addition (ADD)	Pulse With Modulation (PWM)
Bitwise AND (AND)	PWM Frequency (PWM_FREQ)
Average (AVG)	Rising Edge Detect (R_TRIG)
Bit Pack (BIT_PACK)	Convert to Real (REAL)
Bit Unpack (BIT_UNPACK)	Rotate Left (ROL)
Convert to Boolean (BOOLEAN)	Rotate Right (ROR)
Compare (CMP)	Reset / Set -Reset Dominant (RS)
Hardware Counter (CNTRTMR)	Select (SEL)
Count Down (CTD)	Set Date (SETDATE)
Count Up (CTU)	Set Time (SETTIME)
Count Up / Down (CTUD)	Shift Left (SHL)
Division (DIV)	Shift Right (SHR)
Drum Sequencer (DRUM_SEQ)	Serial Print (SERIAL_PRINT)
Falling Edge Detect (F_TRIG)	Set / Reset -Set Dominant (SR)
Get Date (GETDATE)	Subtraction (SUB)
Get Time (GETTIME)	Convert to Timer (TIMER)
High Speed Timer (HIGH_SPD_TMR)	Time Delay Off (TOF)
Hysteresis (HYSTER)	Time Delay On (TON)
Convert to Integer (INTEGER)	Pulse Timer (TP)
Keypad (KEYPAD)	Unlatching Coil (UNLATCH)
Latching Coil (LATCH)	Bitwise XOR (XOR)

### PLC ON A CHIP (PLCHIP-M2-2562X / PLCHIP-M2-2563X)

The supported features include all features that are supported individually. Using certain features may limit availability of other features.

#### SUPPORTED FEATURES:

Analog Inputs ( 8 Channels)	Hardware Counter	Retentive Variables
Real Time Clock (External)	HDIO Bus	Keypad Support
LCD Display Support	PWM Outputs	SPI Slave
Synchronous Serial Interface (SSI)	Serial Printing	Modbus Slave
J1939 Communications	OptiCAN Networking	EEPROM Storage

#### SUPPORTED FUNCTIONS:

Less Than (<)	LCD Print (LCD_PRINT)
Less Than Equal To (<=)	Limit (LIMIT)
Not Equal To (<>)	Moving Average (MAVG)
Equal To (=)	Maximum (MAX)
EEPROM Read (EEPROM_READ)	Minimum (MIN)
EEPROM Write (EEPROM_WRITE)	Modulo (MOD)
Greater Than (>)	Multiplication (MULT)
Greater Than Equal To (>=)	OptiCAN Node Status (OPTICAN-NODESTATUS)
Grey Scale Encoder (GC_SSI)	OptiCAN Transmit Message (OPTICAN_TXNETMSG)
Absolute Value (ABS)	Bitwise NOT (NOT)
Addition (ADD)	Bitwise OR (OR)
Bitwise AND (AND)	Pulse With Modulation (PWM)
Average (AVG)	PWM Frequency (PWM_FREQ)
Bit Pack (BIT_PACK)	Rising Edge Detect (R_TRIG)
Bit Unpack (BIT_UNPACK)	Convert to Real (REAL)
Convert to Boolean (BOOLEAN)	Rotate Left (ROL)
Compare (CMP)	Rotate Right (ROR)
Hardware Counter (CNTRTMR)	Reset / Set -Reset Dominant (RS)
Count Down (CTD)	Select (SEL)
Count Up (CTU)	Set Date (SETDATE)
Count Up / Down (CTUD)	Set Time (SETTIME)
Division (DIV)	Shift Left (SHL)
Drum Sequencer (DRUM_SEQ)	Shift Right (SHR)
Falling Edge Detect (F_TRIG)	Serial Print (SERIAL_PRINT)
Get Date (GETDATE)	Set / Reset -Set Dominant (SR)
Get Time (GETTIME)	Subtraction (SUB)
High Speed Timer (HIGH_SPD_TMR)	Convert to Timer (TIMER)
Hysteresis (HYSTER)	Time Delay Off (TOF)
Convert to Integer (INTEGER)	Time Delay On (TON)
J1939 Receive (J1939_SPN)	Pulse Timer (TP)
Keypad (KEYPAD)	Unlatching Coil (UNLATCH)
Latching Coil (LATCH)	Bitwise XOR (XOR)
LCD Clear (LCD_CLEAR)	

### PLC ON A CHIP MODULE (PLCMOD-M2-12800X)

The supported features include all features that are supported individually. Using certain features may limit availability of other features.

#### SUPPORTED FEATURES:

Analog Inputs ( 8 Channels)  
HDIO Bus

Hardware Counter

Retentive Variables

#### SUPPORTED FUNCTIONS:

Less Than (<)  
Less Than Equal To (<=)  
Not Equal To (<=)  
Equal To (=)  
Greater Than (>)  
Greater Than Equal To (>=)  
Absolute Value (ABS)  
Addition (ADD)  
Bitwise AND (AND)  
Average (AVG)  
Bit Pack (BIT\_PACK)  
Bit Unpack (BIT\_UNPACK)  
Convert to Boolean (BOOLEAN)  
Compare (CMP)  
Hardware Counter (CNTRTMR)  
Count Down (CTD)  
Count Up (CTU)  
Count Up / Down (CTUD)  
Division (DIV)  
Drum Sequencer (DRUM\_SEQ)  
Falling Edge Detect (F\_TRIG)  
High Speed Timer (HIGH\_SPD\_TMR)  
Hysteresis (HYSTER)  
Convert to Integer (INTEGER)  
Latching Coil (LATCH)

Limit (LIMIT)  
Moving Average (MAVG)  
Maximum (MAX)  
Minimum (MIN)  
Modulo (MOD)  
Multiplication (MULT)  
Bitwise NOT (NOT)  
Bitwise OR (OR)  
Rising Edge Detect (R\_TRIG)  
Convert to Real (REAL)  
Rotate Left (ROL)  
Rotate Right (ROR)  
Reset / Set -Reset Dominant (RS)  
Select (SEL)  
Shift Left (SHL)  
Shift Right (SHR)  
Set / Reset -Set Dominant (SR)  
Subtraction (SUB)  
Convert to Timer (TIMER)  
Time Delay Off (TOF)  
Time Delay On (TON)  
Pulse Timer (TP)  
Unlatching Coil (UNLATCH)  
Bitwise XOR (XOR)

### PLC ON A CHIP MODULE (PLCMOD-M2-12801X)

The supported features include all features that are supported individually. Using certain features may limit availability of other features.

#### SUPPORTED FEATURES:

Analog Inputs ( 8 Channels)  
Real Time Clock

Hardware Counter  
HDIO Bus

Retentive Variables

#### SUPPORTED FUNCTIONS:

Less Than (<)  
Less Than Equal To (<=)  
Not Equal To (<=)  
Equal To (=)  
Greater Than (>)  
Greater Than Equal To (>=)  
Absolute Value (ABS)  
Addition (ADD)  
Bitwise AND (AND)  
Average (AVG)  
Bit Pack (BIT\_PACK)  
Bit Unpack (BIT\_UNPACK)  
Convert to Boolean (BOOLEAN)  
Compare (CMP)  
Hardware Counter (CNTRTMR)  
Count Down (CTD)  
Count Up (CTU)  
Count Up / Down (CTUD)  
Division (DIV)  
Drum Sequencer (DRUM\_SEQ)  
Falling Edge Detect (F\_TRIG)  
Get Date (GETDATE)  
Get Time (GETTIME)  
High Speed Timer (HIGH\_SPD\_TMR)  
Hysteresis (HYSTER)  
Convert to Integer (INTEGER)  
Latching Coil (LATCH)

Limit (LIMIT)  
Moving Average (MAVG)  
Maximum (MAX)  
Minimum (MIN)  
Modulo (MOD)  
Multiplication (MULT)  
Bitwise NOT (NOT)  
Bitwise OR (OR)  
Rising Edge Detect (R\_TRIG)  
Convert to Real (REAL)  
Rotate Left (ROL)  
Rotate Right (ROR)  
Reset / Set -Reset Dominant (RS)  
Select (SEL)  
Set Date (SETDATE)  
Set Time (SETTIME)  
Shift Left (SHL)  
Shift Right (SHR)  
Set / Reset -Set Dominant (SR)  
Subtraction (SUB)  
Convert to Timer (TIMER)  
Time Delay Off (TOF)  
Time Delay On (TON)  
Pulse Timer (TP)  
Unlatching Coil (UNLATCH)  
Bitwise XOR (XOR)

### PLC ON A CHIP MODULE (PLCMOD-M2-25600X)

The supported features include all features that are supported individually. Using certain features may limit availability of other features.

#### SUPPORTED FEATURES:

Analog Inputs ( 8 Channels)	Hardware Counter	Retentive Variables
HDIO Bus	Keypad Support	Modbus Slave
LCD Display Support	PWM Outputs	SPI Slave
Synchronous Serial Interface (SSI)	Serial Printing	EEPROM Storage

#### SUPPORTED FUNCTIONS:

Less Than (<)	LCD Clear (LCD_CLEAR)
Less Than Equal To (<=)	LCD Print (LCD_PRINT)
Not Equal To (<>)	Limit (LIMIT)
Equal To (=)	Moving Average (MAVG)
EEPROM Read (EEPROM_READ)	Maximum (MAX)
EEPROM Write (EEPROM_WRITE)	Minimum (MIN)
Greater Than (>)	Modulo (MOD)
Greater Than Equal To (>=)	Multiplication (MULT)
Grey Scale Encoder (GC_SSI)	Bitwise NOT (NOT)
Absolute Value (ABS)	Bitwise OR (OR)
Addition (ADD)	Pulse With Modulation (PWM)
Bitwise AND (AND)	PWM Frequency (PWM_FREQ)
Average (AVG)	Rising Edge Detect (R_TRIG)
Bit Pack (BIT_PACK)	Convert to Real (REAL)
Bit Unpack (BIT_UNPACK)	Rotate Left (ROL)
Convert to Boolean (BOOLEAN)	Rotate Right (ROR)
Compare (CMP)	Reset / Set -Reset Dominant (RS)
Hardware Counter (CNTRTMR)	Select (SEL)
Count Down (CTD)	Shift Left (SHL)
Count Up (CTU)	Shift Right (SHR)
Count Up / Down (CTUD)	Serial Print (SERIAL_PRINT)
Division (DIV)	Set / Reset -Set Dominant (SR)
Drum Sequencer (DRUM_SEQ)	Subtraction (SUB)
Falling Edge Detect (F_TRIG)	Convert to Timer (TIMER)
High Speed Timer (HIGH_SPD_TMR)	Time Delay Off (TOF)
Hysteresis (HYSTER)	Time Delay On (TON)
Convert to Integer (INTEGER)	Pulse Timer (TP)
Keypad (KEYPAD)	Unlatching Coil (UNLATCH)
Latching Coil (LATCH)	Bitwise XOR (XOR)

### PLC ON A CHIP MODULE (PLCMOD-M2-25601X)

The supported features include all features that are supported individually. Using certain features may limit availability of other features.

#### SUPPORTED FEATURES:

Analog Inputs ( 8 Channels)	Hardware Counter	Retentive Variables
Real Time Clock (External)	HDIO Bus	Keypad Support
LCD Display Support	PWM Outputs	SPI Slave
Synchronous Serial Interface (SSI)	Serial Printing	Modbus Slave
EEPROM Storage		

#### SUPPORTED FUNCTIONS:

Less Than (<)	LCD Clear (LCD_CLEAR)
Less Than Equal To (<=)	LCD Print (LCD_PRINT)
Not Equal To (<>)	Limit (LIMIT)
Equal To (=)	Moving Average (MAVG)
EEPROM Read (EEPROM_READ)	Maximum (MAX)
EEPROM Write (EEPROM_WRITE)	Minimum (MIN)
Greater Than (>)	Modulo (MOD)
Greater Than Equal To (>=)	Multiplication (MULT)
Grey Scale Encoder (GC_SSI)	Bitwise NOT (NOT)
Absolute Value (ABS)	Bitwise OR (OR)
Addition (ADD)	Pulse With Modulation (PWM)
Bitwise AND (AND)	PWM Frequency (PWM_FREQ)
Average (AVG)	Rising Edge Detect (R_TRIG)
Bit Pack (BIT_PACK)	Convert to Real (REAL)
Bit Unpack (BIT_UNPACK)	Rotate Left (ROL)
Convert to Boolean (BOOLEAN)	Rotate Right (ROR)
Compare (CMP)	Reset / Set -Reset Dominant (RS)
Hardware Counter (CNTRTMR)	Select (SEL)
Count Down (CTD)	Set Date (SETDATE)
Count Up (CTU)	Set Time (SETTIME)
Count Up / Down (CTUD)	Shift Left (SHL)
Division (DIV)	Shift Right (SHR)
Drum Sequencer (DRUM_SEQ)	Serial Print (SERIAL_PRINT)
Falling Edge Detect (F_TRIG)	Set / Reset -Set Dominant (SR)
Get Date (GETDATE)	Subtraction (SUB)
Get Time (GETTIME)	Convert to Timer (TIMER)
High Speed Timer (HIGH_SPD_TMR)	Time Delay Off (TOF)
Hysteresis (HYSTER)	Time Delay On (TON)
Convert to Integer (INTEGER)	Pulse Timer (TP)
Keypad (KEYPAD)	Unlatching Coil (UNLATCH)
Latching Coil (LATCH)	Bitwise XOR (XOR)

### PLC ON A CHIP MODULE (PLCMOD-M2-25620X / PLCMOD-M2-25630X)

The supported features include all features that are supported individually. Using certain features may limit availability of other features.

#### SUPPORTED FEATURES:

Analog Inputs ( 8 Channels)	Hardware Counter	Retentive Variables
HDIO Bus	Keypad Support	OptiCAN Networking
LCD Display Support	PWM Outputs	SPI Slave
Synchronous Serial Interface (SSI)	Serial Printing	Modbus Slave
J1939 Communications	EEPROM Storage	

#### SUPPORTED FUNCTIONS:

Less Than (<)	LCD Print (LCD_PRINT)
Less Than Equal To (<=)	Limit (LIMIT)
Not Equal To (<=)	Moving Average (MAVG)
Equal To (=)	Maximum (MAX)
EEPROM Read (EEPROM_READ)	Minimum (MIN)
EEPROM Write (EEPROM_WRITE)	Modulo (MOD)
Greater Than (>)	Multiplication (MULT)
Greater Than Equal To (>=)	OptiCAN Node Status (OPTICAN-NODESTATUS)
Grey Scale Encoder (GC_SSI)	OptiCAN Transmit Message (OPTICAN_TXNETMSG)
Absolute Value (ABS)	Bitwise NOT (NOT)
Addition (ADD)	Bitwise OR (OR)
Bitwise AND (AND)	Pulse With Modulation (PWM)
Average (AVG)	PWM Frequency (PWM_FREQ)
Bit Pack (BIT_PACK)	Rising Edge Detect (R_TRIG)
Bit Unpack (BIT_UNPACK)	Convert to Real (REAL)
Convert to Boolean (BOOLEAN)	Rotate Left (ROL)
Compare (CMP)	Rotate Right (ROR)
Hardware Counter (CNTRTMR)	Reset / Set -Reset Dominant (RS)
Count Down (CTD)	Select (SEL)
Count Up (CTU)	Shift Left (SHL)
Count Up / Down (CTUD)	Shift Right (SHR)
Division (DIV)	Serial Print (SERIAL_PRINT)
Drum Sequencer (DRUM_SEQ)	Set / Reset -Set Dominant (SR)
Falling Edge Detect (F_TRIG)	Subtraction (SUB)
High Speed Timer (HIGH_SPD_TMR)	Convert to Timer (TIMER)
Hysteresis (HYSTER)	Time Delay Off (TOF)
Convert to Integer (INTEGER)	Time Delay On (TON)
J1939 Receive (J1939_SPN)	Pulse Timer (TP)
Keypad (KEYPAD)	Unlatching Coil (UNLATCH)
Latching Coil (LATCH)	Bitwise XOR (XOR)
LCD Clear (LCD_CLEAR)	

### PLC ON A CHIP MODULE (PLCMOD-M2-25621X / PLCMOD-M2-25631X)

The supported features include all features that are supported individually. Using certain features may limit availability of other features.

#### SUPPORTED FEATURES:

Analog Inputs ( 8 Channels)	Hardware Counter	Retentive Variables
Real Time Clock (External)	HDIO Bus	Keypad Support
LCD Display Support	PWM Outputs	SPI Slave
Synchronous Serial Interface (SSI)	Serial Printing	Modbus Slave
J1939 Communications	OptiCAN Networking	EEPROM Storage

#### SUPPORTED FUNCTIONS:

Less Than (<)	LCD Print (LCD_PRINT)
Less Than Equal To (<=)	Limit (LIMIT)
Not Equal To (<>)	Moving Average (MAVG)
Equal To (=)	Maximum (MAX)
EEPROM Read (EEPROM_READ)	Minimum (MIN)
EEPROM Write (EEPROM_WRITE)	Modulo (MOD)
Greater Than (>)	Multiplication (MULT)
Greater Than Equal To (>=)	OptiCAN Node Status (OPTICAN-NODESTATUS)
Grey Scale Encoder (GC_SSI)	OptiCAN Transmit Message (OPTICAN_TXNETMSG)
Absolute Value (ABS)	Bitwise NOT (NOT)
Addition (ADD)	Bitwise OR (OR)
Bitwise AND (AND)	Pulse With Modulation (PWM)
Average (AVG)	PWM Frequency (PWM_FREQ)
Bit Pack (BIT_PACK)	Rising Edge Detect (R_TRIG)
Bit Unpack (BIT_UNPACK)	Convert to Real (REAL)
Convert to Boolean (BOOLEAN)	Rotate Left (ROL)
Compare (CMP)	Rotate Right (ROR)
Hardware Counter (CNTRTMR)	Reset / Set -Reset Dominant (RS)
Count Down (CTD)	Select (SEL)
Count Up (CTU)	Set Date (SETDATE)
Count Up / Down (CTUD)	Set Time (SETTIME)
Division (DIV)	Shift Left (SHL)
Drum Sequencer (DRUM_SEQ)	Shift Right (SHR)
Falling Edge Detect (F_TRIG)	Serial Print (SERIAL_PRINT)
Get Date (GETDATE)	Set / Reset -Set Dominant (SR)
Get Time (GETTIME)	Subtraction (SUB)
High Speed Timer (HIGH_SPD_TMR)	Convert to Timer (TIMER)
Hysteresis (HYSTER)	Time Delay Off (TOF)
Convert to Integer (INTEGER)	Time Delay On (TON)
J1939 Receive (J1939_SPN)	Pulse Timer (TP)
Keypad (KEYPAD)	Unlatching Coil (UNLATCH)
Latching Coil (LATCH)	Bitwise XOR (XOR)
LCD Clear (LCD_CLEAR)	



### ENHANCED BABY BEAR (ICM-EBB-100)

The supported features include all features that are supported individually. Using certain features may limit availability of other features.

#### SUPPORTED FEATURES:

Retentive Variables

EEPROM Storage

#### SUPPORTED FUNCTIONS:

Less Than (<)	Latching Coil (LATCH)
Less Than Equal To (<=)	Limit (LIMIT)
Not Equal To (<=)	Moving Average (MAVG)
Equal To (=)	Maximum (MAX)
EEPROM Read (EEPROM_READ)	Minimum (MIN)
EEPROM Write (EEPROM_WRITE)	Modulo (MOD)
Greater Than (>)	Multiplication (MULT)
Greater Than Equal To (>=)	Bitwise NOT (NOT)
Absolute Value (ABS)	Bitwise OR (OR)
Addition (ADD)	Rising Edge Detect (R_TRIG)
Bitwise AND (AND)	Convert to Real (REAL)
Average (AVG)	Rotate Left (ROL)
Bit Pack (BIT_PACK)	Rotate Right (ROR)
Bit Unpack (BIT_UNPACK)	Reset / Set -Reset Dominant (RS)
Convert to Boolean (BOOLEAN)	Select (SEL)
Compare (CMP)	Shift Left (SHL)
Count Down (CTD)	Shift Right (SHR)
Count Up (CTU)	Set / Reset -Set Dominant (SR)
Count Up / Down (CTUD)	Subtraction (SUB)
Division (DIV)	Convert to Timer (TIMER)
Drum Sequencer (DRUM_SEQ)	Time Delay Off (TOF)
Falling Edge Detect (F_TRIG)	Time Delay On (TON)
High Speed Timer (HIGH_SPD_TMR)	Pulse Timer (TP)
Hysteresis (HYSTER)	Unlatching Coil (UNLATCH)
Convert to Integer (INTEGER)	Bitwise XOR (XOR)

### ENHANCED BABY BEAR (ICM-EBB-200)

The supported features include all features that are supported individually. Using certain features may limit availability of other features.

#### SUPPORTED FEATURES:

Retentive Variables

Hardware Counter

EEPROM Storage

#### SUPPORTED FUNCTIONS:

Less Than (<)	Latching Coil (LATCH)
Less Than Equal To (<=)	Limit (LIMIT)
Not Equal To (<=)	Moving Average (MAVG)
Equal To (=)	Maximum (MAX)
EEPROM Read (EEPROM_READ)	Minimum (MIN)
EEPROM Write (EEPROM_WRITE)	Modulo (MOD)
Greater Than (>)	Multiplication (MULT)
Greater Than Equal To (>=)	Bitwise NOT (NOT)
Absolute Value (ABS)	Bitwise OR (OR)
Addition (ADD)	Rising Edge Detect (R_TRIG)
Bitwise AND (AND)	Convert to Real (REAL)
Average (AVG)	Rotate Left (ROL)
Bit Pack (BIT_PACK)	Rotate Right (ROR)
Bit Unpack (BIT_UNPACK)	Reset / Set -Reset Dominant (RS)
Convert to Boolean (BOOLEAN)	Select (SEL)
Compare (CMP)	Shift Left (SHL)
Count Down (CTD)	Shift Right (SHR)
Count Up (CTU)	Set / Reset -Set Dominant (SR)
Count Up / Down (CTUD)	Subtraction (SUB)
Division (DIV)	Convert to Timer (TIMER)
Drum Sequencer (DRUM_SEQ)	Time Delay Off (TOF)
Falling Edge Detect (F_TRIG)	Time Delay On (TON)
Hardware Counter (CNTRTMR)	Pulse Timer (TP)
High Speed Timer (HIGH_SPD_TMR)	Unlatching Coil (UNLATCH)
Hysteresis (HYSTER)	Bitwise XOR (XOR)
Convert to Integer (INTEGER)	

### ENHANCED BABY BEAR (ICM-EBB-300)

The supported features include all features that are supported individually. Using certain features may limit availability of other features.

#### SUPPORTED FEATURES:

Retentive Variables  
Real Time Clock

Hardware Counter

EEPROM Storage

#### SUPPORTED FUNCTIONS:

Less Than (<)	Latching Coil (LATCH)
Less Than Equal To (<=)	Limit (LIMIT)
Not Equal To (<=)	Moving Average (MAVG)
Equal To (=)	Maximum (MAX)
EEPROM Read (EEPROM_READ)	Minimum (MIN)
EEPROM Write (EEPROM_WRITE)	Modulo (MOD)
Greater Than (>)	Multiplication (MULT)
Greater Than Equal To (>=)	Bitwise NOT (NOT)
Absolute Value (ABS)	Bitwise OR (OR)
Addition (ADD)	Rising Edge Detect (R_TRIG)
Bitwise AND (AND)	Convert to Real (REAL)
Average (AVG)	Rotate Left (ROL)
Bit Pack (BIT_PACK)	Rotate Right (ROR)
Bit Unpack (BIT_UNPACK)	Reset / Set -Reset Dominant (RS)
Convert to Boolean (BOOLEAN)	Select (SEL)
Compare (CMP)	Set Date (SETDATE)
Count Down (CTD)	Set Time (SETTIME)
Count Up (CTU)	Shift Left (SHL)
Count Up / Down (CTUD)	Shift Right (SHR)
Division (DIV)	Set / Reset -Set Dominant (SR)
Drum Sequencer (DRUM_SEQ)	Subtraction (SUB)
Falling Edge Detect (F_TRIG)	Convert to Timer (TIMER)
Get Date (GETDATE)	Time Delay Off (TOF)
Get Time (GETTIME)	Time Delay On (TON)
Hardware Counter (CNTRTMR)	Pulse Timer (TP)
High Speed Timer (HIGH_SPD_TMR)	Unlatching Coil (UNLATCH)
Hysteresis (HYSTER)	Bitwise XOR (XOR)
Convert to Integer (INTEGER)	

### ENHANCED BABY BEAR (ICM-EBB-400)

The supported features include all features that are supported individually. Using certain features may limit availability of other features.

#### SUPPORTED FEATURES:

Retentive Variables  
Real Time Clock

Hardware Counter  
Enhanced Baby Bear Expansion Port

EEPROM Storage

#### SUPPORTED FUNCTIONS:

Less Than (<)	Latching Coil (LATCH)
Less Than Equal To (<=)	Limit (LIMIT)
Not Equal To (<>)	Moving Average (MAVG)
Equal To (=)	Maximum (MAX)
EEPROM Read (EEPROM_READ)	Minimum (MIN)
EEPROM Write (EEPROM_WRITE)	Modulo (MOD)
Greater Than (>)	Multiplication (MULT)
Greater Than Equal To (>=)	Bitwise NOT (NOT)
Absolute Value (ABS)	Bitwise OR (OR)
Addition (ADD)	Rising Edge Detect (R_TRIG)
Bitwise AND (AND)	Convert to Real (REAL)
Average (AVG)	Rotate Left (ROL)
Bit Pack (BIT_PACK)	Rotate Right (ROR)
Bit Unpack (BIT_UNPACK)	Reset / Set -Reset Dominant (RS)
Convert to Boolean (BOOLEAN)	Select (SEL)
Compare (CMP)	Set Date (SETDATE)
Count Down (CTD)	Set Time (SETTIME)
Count Up (CTU)	Shift Left (SHL)
Count Up / Down (CTUD)	Shift Right (SHR)
Division (DIV)	Set / Reset -Set Dominant (SR)
Drum Sequencer (DRUM_SEQ)	Subtraction (SUB)
Falling Edge Detect (F_TRIG)	Convert to Timer (TIMER)
Get Date (GETDATE)	Time Delay Off (TOF)
Get Time (GETTIME)	Time Delay On (TON)
Hardware Counter (CNTRTMR)	Pulse Timer (TP)
High Speed Timer (HIGH_SPD_TMR)	Unlatching Coil (UNLATCH)
Hysteresis (HYSTER)	Bitwise XOR (XOR)
Convert to Integer (INTEGER)	

### ENHANCED BABY BEAR (ICM-EBB-500)

The supported features include all features that are supported individually. Using certain features may limit availability of other features.

#### SUPPORTED FEATURES:

Retentive Variables  
Real Time Clock

Hardware Counter  
HDIO Expansion Port

EEPROM Storage

#### SUPPORTED FUNCTIONS:

Less Than (<)  
Less Than Equal To (<=)  
Not Equal To (<=)  
Equal To (=)  
EEPROM Read (EEPROM\_READ)  
EEPROM Write (EEPROM\_WRITE)  
Greater Than (>)  
Greater Than Equal To (>=)  
Absolute Value (ABS)  
Addition (ADD)  
Bitwise AND (AND)  
Average (AVG)  
Bit Pack (BIT\_PACK)  
Bit Unpack (BIT\_UNPACK)  
Convert to Boolean (BOOLEAN)  
Compare (CMP)  
Count Down (CTD)  
Count Up (CTU)  
Count Up / Down (CTUD)  
Division (DIV)  
Drum Sequencer (DRUM\_SEQ)  
Falling Edge Detect (F\_TRIG)  
Get Date (GETDATE)  
Get Time (GETTIME)  
Hardware Counter (CNTRTMR)  
High Speed Timer (HIGH\_SPD\_TMR)  
Hysteresis (HYSTER)  
Convert to Integer (INTEGER)

Latching Coil (LATCH)  
Limit (LIMIT)  
Moving Average (MAVG)  
Maximum (MAX)  
Minimum (MIN)  
Modulo (MOD)  
Multiplication (MULT)  
Bitwise NOT (NOT)  
Bitwise OR (OR)  
Rising Edge Detect (R\_TRIG)  
Convert to Real (REAL)  
Rotate Left (ROL)  
Rotate Right (ROR)  
Reset / Set -Reset Dominant (RS)  
Select (SEL)  
Set Date (SETDATE)  
Set Time (SETTIME)  
Shift Left (SHL)  
Shift Right (SHR)  
Set / Reset -Set Dominant (SR)  
Subtraction (SUB)  
Convert to Timer (TIMER)  
Time Delay Off (TOF)  
Time Delay On (TON)  
Pulse Timer (TP)  
Unlatching Coil (UNLATCH)  
Bitwise XOR (XOR)

### ENHANCED BABY BEAR (ICM-EBB-600)

The supported features include all features that are supported individually. Using certain features may limit availability of other features.

#### SUPPORTED FEATURES:

Retentive Variables	Hardware Counter	EEPROM Storage
Real Time Clock	Enhanced Baby Bear Expansion Port	J1939 Communications
OptiCAN Networking	Modbus Slave	Serial Printing

#### SUPPORTED FUNCTIONS:

Less Than (<)	Limit (LIMIT)
Less Than Equal To (<=)	Moving Average (MAVG)
Not Equal To (<=)	Maximum (MAX)
Equal To (=)	Minimum (MIN)
EEPROM Read (EEPROM_READ)	Modulo (MOD)
EEPROM Write (EEPROM_WRITE)	Multiplication (MULT)
Greater Than (>)	OptiCAN Node Status (OPTICAN-NODESTATUS)
Greater Than Equal To (>=)	OptiCAN Transmit Message (OPTICAN_TXNETMSG)
Absolute Value (ABS)	Bitwise NOT (NOT)
Addition (ADD)	Bitwise OR (OR)
Bitwise AND (AND)	Rising Edge Detect (R_TRIG)
Average (AVG)	Convert to Real (REAL)
Bit Pack (BIT_PACK)	Rotate Left (ROL)
Bit Unpack (BIT_UNPACK)	Rotate Right (ROR)
Convert to Boolean (BOOLEAN)	Reset / Set -Reset Dominant (RS)
Compare (CMP)	Select (SEL)
Hardware Counter (CNTRTMR)	Set Date (SETDATE)
Count Down (CTD)	Set Time (SETTIME)
Count Up (CTU)	Shift Left (SHL)
Count Up / Down (CTUD)	Shift Right (SHR)
Division (DIV)	Serial Print (SERIAL_PRINT)
Drum Sequencer (DRUM_SEQ)	Set / Reset -Set Dominant (SR)
Falling Edge Detect (F_TRIG)	Subtraction (SUB)
Get Date (GETDATE)	Convert to Timer (TIMER)
Get Time (GETTIME)	Time Delay Off (TOF)
High Speed Timer (HIGH_SPD_TMR)	Time Delay On (TON)
Hysteresis (HYSTER)	Pulse Timer (TP)
Convert to Integer (INTEGER)	Unlatching Coil (UNLATCH)
J1939 Receive (J1939_SPN)	Bitwise XOR (XOR)
Latching Coil (LATCH)	

### ENHANCED BABY BEAR (ICM-EBB-700)

The supported features include all features that are supported individually. Using certain features may limit availability of other features.

#### SUPPORTED FEATURES:

Retentive Variables	Hardware Counter	EEPROM Storage
Real Time Clock	HDIO Expansion Port	J1939 Communications
OptiCAN Networking	Modbus Slave	Serial Printing

#### SUPPORTED FUNCTIONS:

Less Than (<)	Limit (LIMIT)
Less Than Equal To (<=)	Moving Average (MAVG)
Not Equal To (<>)	Maximum (MAX)
Equal To (=)	Minimum (MIN)
EEPROM Read (EEPROM_READ)	Modulo (MOD)
EEPROM Write (EEPROM_WRITE)	Multiplication (MULT)
Greater Than (>)	OptiCAN Node Status (OPTICAN-NODESTATUS)
Greater Than Equal To (>=)	OptiCAN Transmit Message (OPTICAN_TXNETMSG)
Absolute Value (ABS)	Bitwise NOT (NOT)
Addition (ADD)	Bitwise OR (OR)
Bitwise AND (AND)	Rising Edge Detect (R_TRIG)
Average (AVG)	Convert to Real (REAL)
Bit Pack (BIT_PACK)	Rotate Left (ROL)
Bit Unpack (BIT_UNPACK)	Rotate Right (ROR)
Convert to Boolean (BOOLEAN)	Reset / Set -Reset Dominant (RS)
Compare (CMP)	Select (SEL)
Hardware Counter (CNTRTMR)	Set Date (SETDATE)
Count Down (CTD)	Set Time (SETTIME)
Count Up (CTU)	Shift Left (SHL)
Count Up / Down (CTUD)	Shift Right (SHR)
Division (DIV)	Serial Print (SERIAL_PRINT)
Drum Sequencer (DRUM_SEQ)	Set / Reset -Set Dominant (SR)
Falling Edge Detect (F_TRIG)	Subtraction (SUB)
Get Date (GETDATE)	Convert to Timer (TIMER)
Get Time (GETTIME)	Time Delay Off (TOF)
High Speed Timer (HIGH_SPD_TMR)	Time Delay On (TON)
Hysteresis (HYSTER)	Pulse Timer (TP)
Convert to Integer (INTEGER)	Unlatching Coil (UNLATCH)
J1939 Receive (J1939_SPN)	Bitwise XOR (XOR)
Latching Coil (LATCH)	

### HARSH ENVIRONMENT CONTROLLER (HEC-1000)

The supported features include all features that are supported individually. Using certain features may limit availability of other features.

#### SUPPORTED FEATURES:

Analog Inputs ( 2 Channels)  
Real Time Clock  
OptiCAN Networking  
Output Monitoring

Hardware Counters (2 Channels)  
PWM Outputs  
EEPROM Storage

Retentive Variables  
J1939 Communications  
Programmable Status LED

#### SUPPORTED FUNCTIONS:

Less Than (<)  
Less Than Equal To (<=)  
Not Equal To (<>)  
Equal To (=)  
EEPROM Read (EEPROM\_READ)  
EEPROM Write (EEPROM\_WRITE)  
Greater Than (>)  
Greater Than Equal To (>=)  
Absolute Value (ABS)  
Addition (ADD)  
Bitwise AND (AND)  
Average (AVG)  
Bit Pack (BIT\_PACK)  
Bit Unpack (BIT\_UNPACK)  
Convert to Boolean (BOOLEAN)  
Compare (CMP)  
Hardware Counter (CNTRTMR)  
Count Down (CTD)  
Count Up (CTU)  
Count Up / Down (CTUD)  
Division (DIV)  
Drum Sequencer (DRUM\_SEQ)  
Falling Edge Detect (F\_TRIG)  
Get Date (GETDATE)  
Get Time (GETTIME)  
High Speed Timer (HIGH\_SPD\_TMR)  
Hysteresis (HYSTER)  
Convert to Integer (INTEGER)  
J1939 Receive (J1939\_SPN)  
Latching Coil (LATCH)

Limit (LIMIT)  
Moving Average (MAVG)  
Maximum (MAX)  
Minimum (MIN)  
Modulo (MOD)  
Multiplication (MULT)  
OptiCAN Node Status (OPTICAN-NODESTATUS)  
OptiCAN Transmit Message (OPTICAN\_TXNETMSG)  
Bitwise NOT (NOT)  
Bitwise OR (OR)  
Pulse With Modulation (PWM)  
PWM Frequency (PWM\_FREQ)  
Rising Edge Detect (R\_TRIG)  
Convert to Real (REAL)  
Rotate Left (ROL)  
Rotate Right (ROR)  
Reset / Set -Reset Dominant (RS)  
Select (SEL)  
Set Date (SETDATE)  
Set Time (SETTIME)  
Shift Left (SHL)  
Shift Right (SHR)  
Set / Reset -Set Dominant (SR)  
Subtraction (SUB)  
Convert to Timer (TIMER)  
Time Delay Off (TOF)  
Time Delay On (TON)  
Pulse Timer (TP)  
Unlatching Coil (UNLATCH)  
Bitwise XOR (XOR)



### HARSH ENVIRONMENT CONTROLLER (HEC-2000)

The supported features include all features that are supported individually. Using certain features may limit availability of other features.

#### SUPPORTED FEATURES:

8 Inputs  
PWM Outputs  
EEPROM Storage

8 Outputs  
J1939 Communications  
Output Monitoring

Retentive Variables  
OptiCAN Networking  
2 Counter Inputs

#### SUPPORTED FUNCTIONS:

Less Than (<)  
Less Than Equal To (<=)  
Not Equal To (<=)  
Equal To (=)  
EEPROM Read (EEPROM\_READ)  
EEPROM Write (EEPROM\_WRITE)  
Greater Than (>)  
Greater Than Equal To (>=)  
Absolute Value (ABS)  
Addition (ADD)  
Bitwise AND (AND)  
Average (AVG)  
Bit Pack (BIT\_PACK)  
Bit Unpack (BIT\_UNPACK)  
Convert to Boolean (BOOLEAN)  
Compare (CMP)  
Hardware Counter (CNTRTMR)  
Count Down (CTD)  
Count Up (CTU)  
Count Up / Down (CTUD)  
Division (DIV)  
Drum Sequencer (DRUM\_SEQ)  
Falling Edge Detect (F\_TRIG)  
Get Date (GETDATE)  
Get Time (GETTIME)  
High Speed Timer (HIGH\_SPD\_TMR)  
Hysteresis (HYSTER)  
Convert to Integer (INTEGER)  
J1939 Receive (J1939\_SPN)  
Latching Coil (LATCH)

Limit (LIMIT)  
Moving Average (MAVG)  
Maximum (MAX)  
Minimum (MIN)  
Modulo (MOD)  
Multiplication (MULT)  
OptiCAN Node Status (OPTICAN-NODESTATUS)  
OptiCAN Transmit Message (OPTICAN\_TXNETMSG)  
Bitwise NOT (NOT)  
Bitwise OR (OR)  
Pulse With Modulation (PWM)  
PWM Frequency (PWM\_FREQ)  
Rising Edge Detect (R\_TRIG)  
Convert to Real (REAL)  
Rotate Left (ROL)  
Rotate Right (ROR)  
Reset / Set -Reset Dominant (RS)  
Select (SEL)  
Set Date (SETDATE)  
Set Time (SETTIME)  
Shift Left (SHL)  
Shift Right (SHR)  
Set / Reset -Set Dominant (SR)  
Subtraction (SUB)  
Convert to Timer (TIMER)  
Time Delay Off (TOF)  
Time Delay On (TON)  
Pulse Timer (TP)  
Unlatching Coil (UNLATCH)  
Bitwise XOR (XOR)

### PLC ON A CHIP CONTROL SYSTEM (PCS-1X0)

The supported features include all features that are supported individually. Using certain features may limit availability of other features.

#### SUPPORTED FEATURES:

Retentive Variables  
Serial Printing

Real Time Clock  
Modbus Slave

HDIO Bus  
EEPROM Storage

#### SUPPORTED FUNCTIONS:

Less Than (<)	Latching Coil (LATCH)
Less Than Equal To (<=)	Limit (LIMIT)
Not Equal To (<=)	Moving Average (MAVG)
Equal To (=)	Maximum (MAX)
EEPROM Read (EEPROM_READ)	Minimum (MIN)
EEPROM Write (EEPROM_WRITE)	Modulo (MOD)
Greater Than (>)	Multiplication (MULT)
Greater Than Equal To (>=)	Bitwise NOT (NOT)
Grey Scale Encoder (GC_SSI)	Bitwise OR (OR)
Absolute Value (ABS)	Rising Edge Detect (R_TRIG)
Addition (ADD)	Convert to Real (REAL)
Bitwise AND (AND)	Rotate Left (ROL)
Average (AVG)	Rotate Right (ROR)
Bit Pack (BIT_PACK)	Reset / Set -Reset Dominant (RS)
Bit Unpack (BIT_UNPACK)	Select (SEL)
Convert to Boolean (BOOLEAN)	Set Date (SETDATE)
Compare (CMP)	Set Time (SETTIME)
Count Down (CTD)	Shift Left (SHL)
Count Up (CTU)	Shift Right (SHR)
Count Up / Down (CTUD)	Serial Print (SERIAL_PRINT)
Division (DIV)	Set / Reset -Set Dominant (SR)
Drum Sequencer (DRUM_SEQ)	Subtraction (SUB)
Falling Edge Detect (F_TRIG)	Convert to Timer (TIMER)
Get Date (GETDATE)	Time Delay Off (TOF)
Get Time (GETTIME)	Time Delay On (TON)
High Speed Timer (HIGH_SPD_TMR)	Pulse Timer (TP)
Hysteresis (HYSTER)	Unlatching Coil (UNLATCH)
Convert to Integer (INTEGER)	Bitwise XOR (XOR)

### PLC ON A CHIP CONTROL SYSTEM (PCS-1X1 / PCS-1X2)

The supported features include all features that are supported individually. Using certain features may limit availability of other features.

#### SUPPORTED FEATURES:

Retentive Variables  
Serial Printing  
Analog Inputs (6 Channels)

Real Time Clock  
Modbus Slave  
Analog Outputs (4 Channels)

HDIO Bus  
EEPROM Storage  
PWM Outputs (2 Channels)

#### SUPPORTED FUNCTIONS:

Less Than (<)  
Less Than Equal To (<=)  
Not Equal To (<=)  
Equal To (=)  
EEPROM Read (EEPROM\_READ)  
EEPROM Write (EEPROM\_WRITE)  
Greater Than (>)  
Greater Than Equal To (>=)  
Absolute Value (ABS)  
Addition (ADD)  
Bitwise AND (AND)  
Average (AVG)  
Bit Pack (BIT\_PACK)  
Bit Unpack (BIT\_UNPACK)  
Convert to Boolean (BOOLEAN)  
Compare (CMP)  
Count Down (CTD)  
Count Up (CTU)  
Count Up / Down (CTUD)  
Division (DIV)  
Drum Sequencer (DRUM\_SEQ)  
Falling Edge Detect (F\_TRIG)  
Get Date (GETDATE)  
Get Time (GETTIME)  
High Speed Timer (HIGH\_SPD\_TMR)  
Hysteresis (HYSTER)  
Convert to Integer (INTEGER)  
Latching Coil (LATCH)  
Limit (LIMIT)

Moving Average (MAVG)  
Maximum (MAX)  
Minimum (MIN)  
Modulo (MOD)  
Multiplication (MULT)  
Bitwise NOT (NOT)  
Bitwise OR (OR)  
Pulse With Modulation (PWM)  
PWM Frequency (PWM\_FREQ)  
Rising Edge Detect (R\_TRIG)  
Convert to Real (REAL)  
Rotate Left (ROL)  
Rotate Right (ROR)  
Reset / Set -Reset Dominant (RS)  
Select (SEL)  
Set Date (SETDATE)  
Set Time (SETTIME)  
Shift Left (SHL)  
Shift Right (SHR)  
Serial Print (SERIAL\_PRINT)  
Set / Reset -Set Dominant (SR)  
Subtraction (SUB)  
Convert to Timer (TIMER)  
Time Delay Off (TOF)  
Time Delay On (TON)  
Pulse Timer (TP)  
Unlatching Coil (UNLATCH)  
Bitwise XOR (XOR)

### PLC ON A CHIP CONTROL SYSTEM (PCS-2X0)

The supported features include all features that are supported individually. Using certain features may limit availability of other features.

#### SUPPORTED FEATURES:

Retentive Variables	Real Time Clock	HDIO Bus
Serial Printing	Modbus Slave	EEPROM Storage
Hardware Counter	Grey Scale SSI Encoder Interface	J1939 Communications
OptiCAN Networking		

#### SUPPORTED FUNCTIONS:

Less Than (<)	Latching Coil (LATCH)
Less Than Equal To (<=)	Limit (LIMIT)
Not Equal To (<=)	Moving Average (MAVG)
Equal To (=)	Maximum (MAX)
EEPROM Read (EEPROM_READ)	Minimum (MIN)
EEPROM Write (EEPROM_WRITE)	Modulo (MOD)
Greater Than (>)	Multiplication (MULT)
Greater Than Equal To (>=)	OptiCAN Node Status (OPTICAN-NODESTATUS)
Grey Scale Encoder (GC_SSI)	OptiCAN Transmit Message (OPTICAN_TXNETMSG)
Absolute Value (ABS)	Bitwise NOT (NOT)
Addition (ADD)	Bitwise OR (OR)
Bitwise AND (AND)	Rising Edge Detect (R_TRIG)
Average (AVG)	Convert to Real (REAL)
Bit Pack (BIT_PACK)	Rotate Left (ROL)
Bit Unpack (BIT_UNPACK)	Rotate Right (ROR)
Convert to Boolean (BOOLEAN)	Reset / Set -Reset Dominant (RS)
Compare (CMP)	Select (SEL)
Hardware Counter (CNTRTMR)	Set Date (SETDATE)
Count Down (CTD)	Set Time (SETTIME)
Count Up (CTU)	Shift Left (SHL)
Count Up / Down (CTUD)	Shift Right (SHR)
Division (DIV)	Serial Print (SERIAL_PRINT)
Drum Sequencer (DRUM_SEQ)	Set / Reset -Set Dominant (SR)
Falling Edge Detect (F_TRIG)	Subtraction (SUB)
Get Date (GETDATE)	Convert to Timer (TIMER)
Get Time (GETTIME)	Time Delay Off (TOF)
High Speed Timer (HIGH_SPD_TMR)	Time Delay On (TON)
Hysteresis (HYSTER)	Pulse Timer (TP)
Convert to Integer (INTEGER)	Unlatching Coil (UNLATCH)
J1939 Receive (J1939_SPN)	Bitwise XOR (XOR)

### PLC ON A CHIP CONTROL SYSTEM (PCS-2X1 / PCS-2X2)

The supported features include all features that are supported individually. Using certain features may limit availability of other features.

#### SUPPORTED FEATURES:

Retentive Variables	Real Time Clock	HDIO Bus
Serial Printing	Modbus Slave	EEPROM Storage
Hardware Counter	Grey Scale SSI Encoder Interface	J1939 Communications
OptiCAN Networking	Analog Inputs (6 Channels)	PWM Outputs (2 Channels)
Analog Outputs (4 Channels)		

#### SUPPORTED FUNCTIONS:

Less Than (<)	Limit (LIMIT)
Less Than Equal To (<=)	Moving Average (MAVG)
Not Equal To (<>)	Maximum (MAX)
Equal To (=)	Minimum (MIN)
EEPROM Read (EEPROM_READ)	Modulo (MOD)
EEPROM Write (EEPROM_WRITE)	Multiplication (MULT)
Greater Than (>)	OptiCAN Node Status (OPTICAN-NODESTATUS)
Greater Than Equal To (>=)	OptiCAN Transmit Message (OPTICAN_TXNETMSG)
Grey Scale Encoder (GC_SSI)	Bitwise NOT (NOT)
Absolute Value (ABS)	Bitwise OR (OR)
Addition (ADD)	Pulse With Modulation (PWM)
Bitwise AND (AND)	PWM Frequency (PWM_FREQ)
Average (AVG)	Rising Edge Detect (R_TRIG)
Bit Pack (BIT_PACK)	Convert to Real (REAL)
Bit Unpack (BIT_UNPACK)	Rotate Left (ROL)
Convert to Boolean (BOOLEAN)	Rotate Right (ROR)
Compare (CMP)	Reset / Set -Reset Dominant (RS)
Hardware Counter (CNTRTMR)	Select (SEL)
Count Down (CTD)	Set Date (SETDATE)
Count Up (CTU)	Set Time (SETTIME)
Count Up / Down (CTUD)	Shift Left (SHL)
Division (DIV)	Shift Right (SHR)
Drum Sequencer (DRUM_SEQ)	Serial Print (SERIAL_PRINT)
Falling Edge Detect (F_TRIG)	Set / Reset -Set Dominant (SR)
Get Date (GETDATE)	Subtraction (SUB)
Get Time (GETTIME)	Convert to Timer (TIMER)
High Speed Timer (HIGH_SPD_TMR)	Time Delay Off (TOF)
Hysteresis (HYSTER)	Time Delay On (TON)
Convert to Integer (INTEGER)	Pulse Timer (TP)
J1939 Receive (J1939_SPN)	Unlatching Coil (UNLATCH)
Latching Coil (LATCH)	Bitwise XOR (XOR)

### SOLVES-It! PLUG IN PLC (SI-100)

The supported features include all features that are supported individually. Using certain features may limit availability of other features.

#### SUPPORTED FEATURES:

EEPROM Storage  
4 Inputs

Programmable LEDs (4)  
4 Outputs

Hardware Counter

#### SUPPORTED FUNCTIONS:

Less Than (<)	Latching Coil (LATCH)
Less Than Equal To (<=)	Limit (LIMIT)
Not Equal To (<=)	Moving Average (MAVG)
Equal To (=)	Maximum (MAX)
EEPROM Read (EEPROM_READ)	Minimum (MIN)
EEPROM Write (EEPROM_WRITE)	Modulo (MOD)
Greater Than (>)	Multiplication (MULT)
Greater Than Equal To (>=)	Bitwise NOT (NOT)
Absolute Value (ABS)	Bitwise OR (OR)
Addition (ADD)	Rising Edge Detect (R_TRIG)
Bitwise AND (AND)	Convert to Real (REAL)
Average (AVG)	Rotate Left (ROL)
Bit Pack (BIT_PACK)	Rotate Right (ROR)
Bit Unpack (BIT_UNPACK)	Reset / Set -Reset Dominant (RS)
Convert to Boolean (BOOLEAN)	Select (SEL)
Compare (CMP)	Shift Left (SHL)
Hardware Counter (CNTRTMR)	Shift Right (SHR)
Count Down (CTD)	Set / Reset -Set Dominant (SR)
Count Up (CTU)	Subtraction (SUB)
Count Up / Down (CTUD)	Convert to Timer (TIMER)
Division (DIV)	Time Delay Off (TOF)
Drum Sequencer (DRUM_SEQ)	Time Delay On (TON)
Falling Edge Detect (F_TRIG)	Pulse Timer (TP)
Hysteresis (HYSTER)	Unlatching Coil (UNLATCH)
Convert to Integer (INTEGER)	Bitwise XOR (XOR)

### SOLVES-It! PLUG IN PLC (SI-200)

The supported features include all features that are supported individually. Using certain features may limit availability of other features.

#### SUPPORTED FEATURES:

EEPROM Storage	4 Digit 7-Segment Display	Programmable LEDs (4)
Programmable Pushbuttons (2)	Real Time Clock	Retentive Variables
Hardware Counter	4 Inputs	4 Outputs

#### SUPPORTED FUNCTIONS:

Less Than (<)	Latching Coil (LATCH)
Less Than Equal To (<=)	Limit (LIMIT)
Not Equal To (<=)	Moving Average (MAVG)
Equal To (=)	Maximum (MAX)
EEPROM Read (EEPROM_READ)	Minimum (MIN)
EEPROM Write (EEPROM_WRITE)	Modulo (MOD)
Greater Than (>)	Multiplication (MULT)
Greater Than Equal To (>=)	Bitwise NOT (NOT)
Absolute Value (ABS)	Bitwise OR (OR)
Addition (ADD)	Rising Edge Detect (R_TRIG)
Bitwise AND (AND)	Convert to Real (REAL)
Average (AVG)	Rotate Left (ROL)
Bit Pack (BIT_PACK)	Rotate Right (ROR)
Bit Unpack (BIT_UNPACK)	Reset / Set -Reset Dominant (RS)
Clear Display (SI_CLRDISP)	Select (SEL)
Convert to Boolean (BOOLEAN)	Set Date (SETDATE)
Compare (CMP)	Set Time (SETTIME)
Hardware Counter (CNTRTMR)	Shift Left (SHL)
Count Down (CTD)	Shift Right (SHR)
Count Up (CTU)	Set / Reset -Set Dominant (SR)
Count Up / Down (CTUD)	Subtraction (SUB)
Division (DIV)	Convert to Timer (TIMER)
Drum Sequencer (DRUM_SEQ)	Time Delay Off (TOF)
Falling Edge Detect (F_TRIG)	Time Delay On (TON)
Get Date (GETDATE)	Pulse Timer (TP)
Get Time (GETTIME)	Unlatching Coil (UNLATCH)
Hysteresis (HYSTER)	Write to Display (SI_DISP)
Convert to Integer (INTEGER)	Bitwise XOR (XOR)

### SOLVES-IT! PLUG IN PLC (SI-110)

The supported features include all features that are supported individually. Using certain features may limit availability of other features.

#### SUPPORTED FEATURES:

EEPROM Storage  
3 Analog Inputs (2 Pots)

Programmable LED (1)  
4 Dual Function I/O (In/Out)

Hardware Counter  
2 Outputs

#### SUPPORTED FUNCTIONS:

Less Than (<)  
Less Than Equal To (<=)  
Not Equal To (<=)  
Equal To (=)  
EEPROM Read (EEPROM\_READ)  
EEPROM Write (EEPROM\_WRITE)  
Greater Than (>)  
Greater Than Equal To (>=)  
Absolute Value (ABS)  
Addition (ADD)  
Bitwise AND (AND)  
Average (AVG)  
Bit Pack (BIT\_PACK)  
Bit Unpack (BIT\_UNPACK)  
Convert to Boolean (BOOLEAN)  
Compare (CMP)  
Hardware Counter (CNTRTMR)  
Count Down (CTD)  
Count Up (CTU)  
Count Up / Down (CTUD)  
Division (DIV)  
Drum Sequencer (DRUM\_SEQ)  
Falling Edge Detect (F\_TRIG)  
Hysteresis (HYSTER)  
Convert to Integer (INTEGER)

Latching Coil (LATCH)  
Limit (LIMIT)  
Moving Average (MAVG)  
Maximum (MAX)  
Minimum (MIN)  
Modulo (MOD)  
Multiplication (MULT)  
Bitwise NOT (NOT)  
Bitwise OR (OR)  
Rising Edge Detect (R\_TRIG)  
Convert to Real (REAL)  
Rotate Left (ROL)  
Rotate Right (ROR)  
Reset / Set -Reset Dominant (RS)  
Select (SEL)  
Shift Left (SHL)  
Shift Right (SHR)  
Set / Reset -Set Dominant (SR)  
Subtraction (SUB)  
Convert to Timer (TIMER)  
Time Delay Off (TOF)  
Time Delay On (TON)  
Pulse Timer (TP)  
Unlatching Coil (UNLATCH)  
Bitwise XOR (XOR)



### SOLVES-It! PLUG IN PLC (SI-210)

The supported features include all features that are supported individually. Using certain features may limit availability of other features.

#### SUPPORTED FEATURES:

EEPROM Storage	4 Digit 7-Segment Display	Programmable LEDs (4)
Programmable Pushbuttons (2)	Real Time Clock	Retentive Variables
Hardware Counter	3 Analog Inputs (2 Pots)	4 Dual Function I/O (In/Out)
2 Outputs		

#### SUPPORTED FUNCTIONS:

Less Than (<)	Latching Coil (LATCH)
Less Than Equal To (<=)	Limit (LIMIT)
Not Equal To (<=)	Moving Average (MAVG)
Equal To (=)	Maximum (MAX)
EEPROM Read (EEPROM_READ)	Minimum (MIN)
EEPROM Write (EEPROM_WRITE)	Modulo (MOD)
Greater Than (>)	Multiplication (MULT)
Greater Than Equal To (>=)	Bitwise NOT (NOT)
Absolute Value (ABS)	Bitwise OR (OR)
Addition (ADD)	Rising Edge Detect (R_TRIG)
Bitwise AND (AND)	Convert to Real (REAL)
Average (AVG)	Rotate Left (ROL)
Bit Pack (BIT_PACK)	Rotate Right (ROR)
Bit Unpack (BIT_UNPACK)	Reset / Set -Reset Dominant (RS)
Clear Display (SI_CLRDISP)	Select (SEL)
Convert to Boolean (BOOLEAN)	Set Date (SETDATE)
Compare (CMP)	Set Time (SETTIME)
Hardware Counter (CNTRTMR)	Shift Left (SHL)
Count Down (CTD)	Shift Right (SHR)
Count Up (CTU)	Set / Reset -Set Dominant (SR)
Count Up / Down (CTUD)	Subtraction (SUB)
Division (DIV)	Convert to Timer (TIMER)
Drum Sequencer (DRUM_SEQ)	Time Delay Off (TOF)
Falling Edge Detect (F_TRIG)	Time Delay On (TON)
Get Date (GETDATE)	Pulse Timer (TP)
Get Time (GETTIME)	Unlatching Coil (UNLATCH)
Hysteresis (HYSTER)	Write to Display (SI_DISP)
Convert to Integer (INTEGER)	Bitwise XOR (XOR)

# SECTION 18

## EZ LADDER REPORTS



EZ LADDER provides helpful reports for project management. These reports are helpful in troubleshooting and documentation upkeep for a project.

### Variable Definitions

The variable definitions report provides a summary of all of the variables in the ladder diagram project. These variables are sorted by type for easy reference. For each variable, the report shows *Name*, *Type*, *I/O Number*, *Default Value* and its *Description*.

To view this report, select **VARIABLE DEFINITIONS** from the **REPORT** menu. Figure 8.1 shows a sample *Variable Definitions Report*.

### Cross References

The Cross Reference Report provides a summary of the objects in the ladder diagram project. These object are sorted by the type of object. The report is customizable.

To view this report, select **CROSS REFERENCES** from the **REPORT** menu. A configuration dialog will open allowing the selection of what to view in the report. Figure 8.2 shows a sample *Cross Refernces Report*.

Input:	Will show all the inputs on the report.
Internal:	Will show all the internal contacts and coils on the report.
Function:	Will show all the functions on the report.
Unused Variables:	Will show any variables declared but not used in the ladder diagram on the report.
Contact without Coil:	Will show any contacts in the ladder diagram that do not have associated coils on the report.
Coil without Contact:	Will show any coils in the ladder diagram that do not have associated contacts on the report.
Drum Sequencer Tables:	Will show the I/O and state maxtrix table for the drum sequencers on the report.

For each of the items selected to add to the report, the report will show the Name, *Rung Number*, *Type* and *Description*.

## Variable Definitions

Date: 1/4/2005 2:33 PM  
Filename: C:\temp\EZ Ladder3.dld  
Target: Divelbiss Corporation PLC-ON-A-CHIP  
Version: 0.0.0.0  
Build Number: 1

### BOOLEAN

Name	Type	I/O Number	Default Value	Description
CR1	INTERNAL		0	
CR2	INTERNAL		0	
CR3	INTERNAL			
CR4	INTERNAL			

### INTEGER

Name	Type	I/O Number	Default Value	Description
cntval	INTERNAL			
count	INTERNAL			

### REAL

Name	Type	I/O Number	Default Value	Description
------	------	------------	---------------	-------------

### TIMER

Name	Type	I/O Number	Default Value	Description
------	------	------------	---------------	-------------

Figure 18.1

## Cross References

Date: 1/4/2005 3:03 PM  
 Filename: C:\temp\EZ Ladder3.dld  
 Target: Divelbiss Corporation PLC-ON-A-CHIP  
 Version: 0.0.0.0  
 Build Number: 1

### INPUT

Name	Rung	Description
------	------	-------------

### OUTPUT

Name	Rung	Description
------	------	-------------

### INTERNAL

Name	Rung	Description
CR1	2	
CR2	3	
CR3	1	
CR4	1	
cntval	3	
count	3	

### FUNCTION

Name	Rung	Type	Description
CTD1	1	CntrCtd	

### NOT USED

### CONTACTS WITHOUT COILS

Name	Rung	Description
CR1	2	
CR2	3	
CR3	1	

### COILS WITHOUT CONTACTS

Name	Rung	Description
CR4	1	

Figure 18.2

# SECTION 19

## TROUBLESHOOTING



## Error Messages

The following is a list and description of errors that may be encountered while using EZ LADDER to develop, download and monitor programs.

**A different program is running** (Monitor Mode)

When connecting to a target, the program running on the target is different than the program currently opened in EZ Ladder.

**Could not connect to target** (Monitor Mode)

EZ Ladder was not able to connect to a hardware target.

**Could not get target version. Please connect first** (Monitor Mode)

EZ Ladder was unable to retrieve the target version when using the “target information” function.

**Could not open: COMX** (Monitor Mode)

When connecting to a target, the selected Com Port does not exist or is in use.

**Error downloading file** (Monitor Mode)

An unknown error occurred while downloading the program to target. Try downloading the program again.

**ERROR downloading user program: invalid address** (Monitor Mode)

An invalid address was detected in a communications packet while EZ Ladder was connected to a target.

**ERROR downloading user program: invalid record** (Monitor Mode)

An invalid record was detected in a communications packet while EZ Ladder was connected to a target.

**ERROR downloading user program: checksum error** (Monitor Mode)

An invalid checksum was detected in a communications packet while EZ Ladder was connected to a target.

**ERROR downloading user program: record too long** (Monitor Mode)

An invalid record length was detected in a communications packet while EZ Ladder was connected to a target.

**ERROR putting target into bootloader** (Monitor Mode)

An error occurred when EZ Ladder was trying to access the target bootloader.

**Error, serial port not open** (Monitor Mode)

The serial port that is configured in EZ Ladder cannot be opened for use.

**ERROR programming target** (Monitor Mode)

EZ Ladder detected an undefined error while attempting to save the program on the target.

**Error starting program. Program doesn't exist** (Monitor Mode)

The program that is trying to start does not exist on the target. Download the program.

**Error starting program. Program could not be started** (Monitor Mode)

The program cannot be started. Re-compile and download the program.

**Error while receiving packet** (Monitor Mode)

There was an error when receiving communications packets from the target.

**File could not be opened** (Monitor Mode)

When downloading the program to target, the file with the compiled code could not be opened.

**Invalid File** (Editor Mode)

The file selected to load was not an EZ Ladder diagram file.

**Invalid HEX file (Monitor Mode)**

When downloading to a target, the file used to store compiled code is invalid, or corrupt. The ladder diagram needs to be re-compiled.

**x is not supported by the current target (Editor Mode)**

The selected object is not supported by specified target.

**Ladder program is not present (Monitor Mode)**

No ladder diagram program was detected on the target connected to EZ Ladder.

**Link at: (x, y) had an invalid Grid point (Editor Mode)****Link is not valid (Editor Mode)**

The link you are trying to create is not valid. Check the object types you trying to connect.

**No acknowledgement sent from target (x) (Monitor Mode)**

The target did not send a no acknowledgement during communications with EZ Ladder.

**Object already there (Editor Mode)**

An object already exists where you are trying to place another object. Select a new location to place the object.

**Object type: X, not found Aborting load (Editor Mode)**

Error loading program into EZ Ladder. The ladder diagram file may be corrupt.

**Packet contained a formatting ERROR (Monitor Mode)**

An packet formatting error was detected in a packet during communication with a target.

**Packet contained an invalid checksum (Monitor Mode)**

An invalid checksum was detected in a packet during communication with a target.

**Packet length was invalid (Monitor Mode)**

EZ Ladder has detected a invalid communications packet length during communications with the connected target.

**Please save project before compiling (Editor Mode)**

Unable to compile ladder diagram, program must have a valid filename.

**Please select a target (Editor Mode)**

A target has not been selected.

**Please select a target before compiling (Editor Mode)**

Unable to compile because no target was selected.

**Please select a target before verifying (Editor Mode)**

Unable to run program verification because no target was selected prior.

**Targets do not match (Monitor Mode)**

When connecting to a target the target specified in the ladder diagram does not match the actual detected hardware target.

**Target does not support bootloader (Monitor Mode)****There is not enough room for the paste. Increase the number of rungs (Editor Mode)**

There is not enough rung space to paste from the clipboard. Increase the number of rungs.



**There is not enough room to the right of the paste point.** (Editor Mode)

There is not enough room at the insertion point to paste objects from the clipboard. Paste the objects farther left.

**This object must be place in the last column** (Editor Mode)

The selected object can only be placed in the last column (coils).

**Timeout ERROR. Entire packet was not received** (Monitor Mode)

During communication with a target, part of a packet was lost or not received.

**Timeout ERROR. Target didn't respond** (Monitor Mode)

During communication with a target, the target did not respond. Check the connections and serial port setting in EZ Ladder.

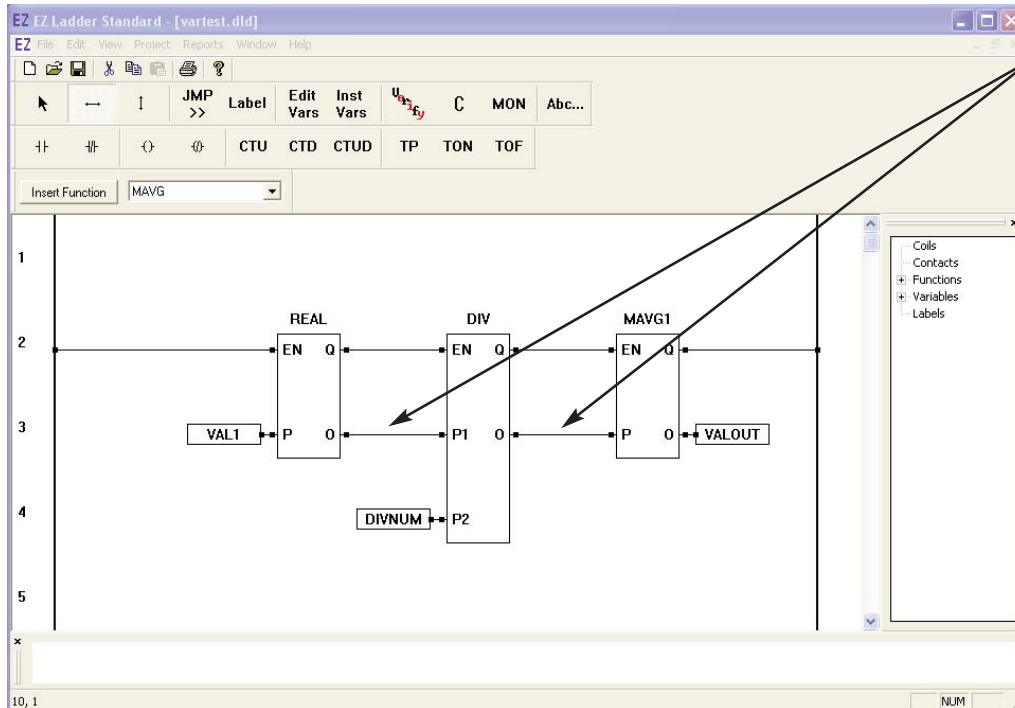
**Undefined packet type** (Monitor Mode)

EZ Ladder has detected a undefined communications packet during communications with the connected target.

## Connecting Functions to Functions Mistakes

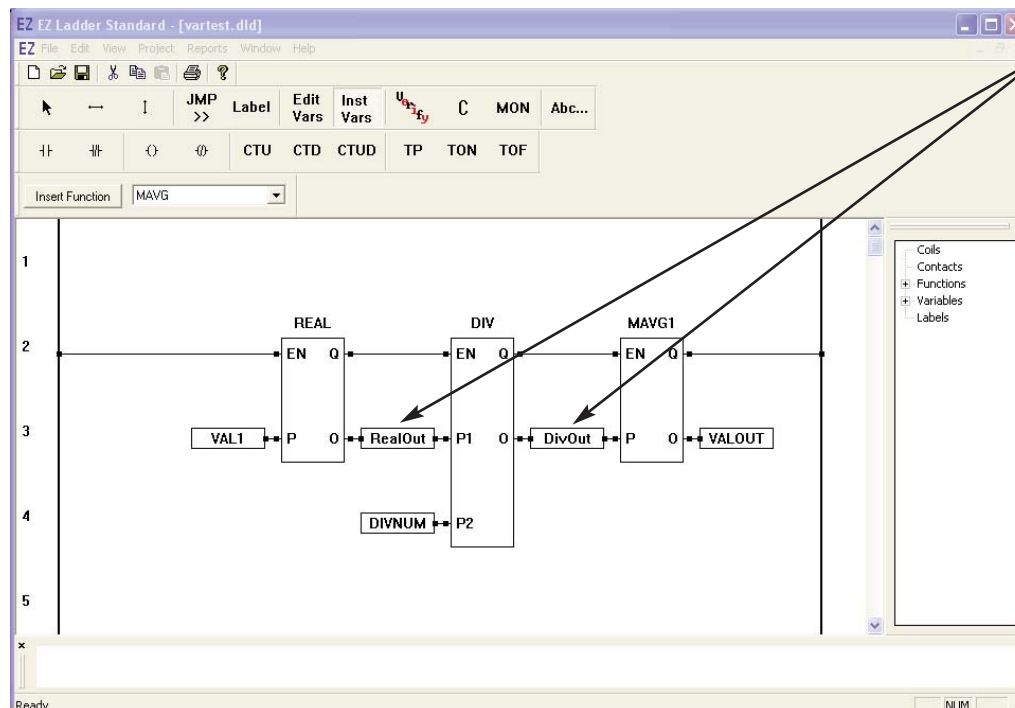
The following apply to using functions with other functions.

1. Variable outputs of a function connected directly to a variable input of another function. When connected directly, the program will compile successfully, however; the program will not function as designed. A variable must be placed between the output and the input for proper operation. Figure 19.1 shows the incorrect way to connect two function's variable I/O. Figure 19.2 shows the correct connection.



**Incorrect - Variables Output connected directly to variable input.**

**Figure 19.1**



**Correct - Function variable output and input connected through inserted variable.**

**Figure 19.2**

# SECTION 20

## EZ LADDER FUNCTIONS & OBJECTS



## Object Basics

This sections provides information on using each of the standard functions and objects using EZ LADDER. For each object or function, the symbol diagram, information on the inputs and outputs and a description of the function or block operation is provided. When applicable, truth tables, timing diagrams or other functions details are provided.

This information is to provide basic practices of how each function or object works and is not intended to provide complete applications or uses.

Availability of functions and objects is determined by the hardware target that is configured for the ladder diagram projects. Some functions and objects are not available on some targets. Refer to the actual hardware target's datasheet or manual for a complete list of supported functions and objects.

## List of EZ Ladder Objects

<u>Name</u>	<u>Description</u>	<u>Page</u>
ABS	Outputs the absolute value of input variable value.	20-4
ACOS	Outputs the arccosine of the input variable value	20-5
ADD	Sums all the input variable values together.	20-6
AND	Bitwise AND function of the input variables.	20-7
ASIN	Outputs the arcsine of the input variable value.	20-8
ATAN	Outputs the arctangent of the input variable value.	20-9
AVG	Averages the input variable values.	20-10
BIT_PACK	Converts up to 32 boolean inputs into a single 32-bit integer.	20-11
BIT_UNPACK	Converts a single 32-bit integer into up to 32 boolean outputs.	20-12
BOOLEAN	Converts input variable value into a Boolean variable.	20-13
CEIL	Outputs a rounded-up value for the input variable value.	20-14
CMP	Compares input variable values.	20-15
CNTRTMR	Function for using hardware timer / counter circuit.	20-16
COS	Outputs the cosine of the input variable value.	20-17
CTD	Down Counter, Counts in the downward direction.	20-18
CTU	Up Counter, Counts in the upward direction	20-19
CTUD	Up/Down Counter, Counts in the Up and Down direction	20-20
DIRECT COIL	Normally De-energized, Boolean Output or Internal Coil.	20-21
DIRECT CONTACT	Normally Open, Boolean Input or Internal Contact	20-22
DIV	Divides by input variable value.	20-23
DRUM_SEQ	Drum Sequencer	20-24
= (EQUAL TO)	Provides equal comparison of variable input values.	20-25
EEPROM READ	Block to read values from EEPROM storage	20-26
EEPROM WRITE	Block to write values to EEPROM storage	20-27
EXP	Outputs Natural Exponential of variable input value.	20-28
EXPT	Outputs Exponentiation for the variable input values.	20-29
F_TRIG	Falling Edge Trigger	20-30
FLOOR	Outputs a rounded-down value for the input variable value.	20-31
GC_SSI	Interface to encoder with gray code synchronous serial	20-32
GETDATE	Reads current date from hardware Real Time Clock	20-34
GETTIME	Reads current time from hardware Real Time Clock	20-35
> (GREATER THAN)	If greater than comparison of variable input values.	20-36
>= (GREATER THAN EQUAL TO)	If greater than equal to comparison of variable input values.	20-37
HIGH_SPD_TMR	100 microsecond resolution Rising to Falling Edge Timer	20-38
HYSTER	Hysteresis	20-39
INTEGER	Converts input variable value into a Integer variable.	20-40
INVERTED COIL	Normally Energized, Boolean Output or Internal Coil.	20-41

<u>Name</u>	<u>Description</u>	<u>Page</u>
INVERTED CONTACT	Normally Closed, Boolean Input or Internal Contact.	20-42
J1939_SPN	Reads broadcast messages from J1939 CAN Network	20-43
JMP	Jump to another section of the program.	20-46
KEYPAD	Reads the Keypad matrix.	20-47
LABEL	Labels sections for the JMP function to operate correctly.	20-49
LATCH (coil)	Direct Coil that maintains it's energized state until UNLATCHed.	20-50
LCD_CLEAR	Clears the LCD Display	20-51
LCD_PRINT	Prints to the LCD Display	20-52
< (LESS THAN)	If less than comparison of variable input values.	20-53
<= (LESS THAN EQUAL TO)	If less than or equal to comparison of variable input values.	20-55
LIMIT	Limits range of a variable.	20-56
LN	Outputs Natural Logarithm of variable input value.	20-57
LOG	Calculates the logarithm (base 10) of the variable input value.	20-58
MAVG	Provides a moving average of the input variable values.	20-59
MAX	Outputs the largest of the input variable values.	20-60
MIN	Outputs the smallest of the input variable values.	20-61
MOD	Calculates the modulo value of the input variable values.	20-62
MULT	Multiplies the input variable values.	20-63
MUX	Provides selection of multiple input variables.	20-64
NOT	One's complement of the input variable value.	20-65
<> (NOT EQUAL TO)	Inequality comparison of input variable values.	20-66
OPTICAN_NODESTATUS	Provides status of an OptiCAN network Node	20-67
OPTICAN_TXNETMSG	Broadcasts Global Start, Stop and Reset for OptiCAN Networks	20-68
OR	Bitwise OR function of the input variables.	20-69
PID	PID Control Algorithm	20-70
PWM	Controls a PWM output channel (hardware)	20-71
PWM_FREQ	Changes / controls the frequency of the PWM clock.	20-72
RANDOM	Outputs a random number based on a SEED value.	20-73
REAL	Converts input variable value into a Real variable.	20-74
R_TRIG	Rising Edge Detector	20-75
ROL	Provides a Left Bit Rotation	20-76
ROR	Provides a Right Bit Rotation	20-77
RS	Reset Dominant bistable.	20-78
SEED	Used to provide the RANDOM with a base number.	20-79
SEL	Selection Block	20-80
SERIAL_PRINT	Serial prints to the multi-pupose port or 2nd COM port.	20-81
SETDATE	Sets the date of the hardware Real Time Clock	20-83
SETTIME	Sets the time of the hardware Real Time Clock	20-84
SHL	Bit Shift Left	20-85
SHR	Bit Shift Right	20-86
SIN	Outputs the sine of the input variable value.	20-87
SI_DISP	Writes to the Solves-it! display (integer only)	20-88
SI_CLRDISP	Clears the Solves-It! display.	20-89
SQRT	Outputs the square root of the input variable value.	20-90
SR	Set Dominant bistable	20-91
SUB	Subtracts input variables values.	20-92
TAN	Outputs the tangent of the input variable value	20-93
TIMER	Converts Integer or Real to Timer	20-94
TOF	Off Delay Timer (Time Delay on Drop Out)	20-95
TON	On Delay Timer (Time Delay on Pick-Up)	20-96
TP	Pulse Timer (one-shot timer)	20-97
UNLATCH (coil)	Direct Coil de-energize a LATCHed coil	20-98
XOR	Bitwise Exclusive OR of the input variables.	20-99

ABS

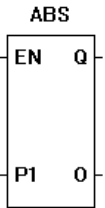
Inputs

- EN Boolean Function Enable Input - function is disabled if EN is false.
- P1 Integer/Real Input - Output is absolute of this input.

Outputs

- Q Boolean Function Enable Output - true when function is enabled.
- O Integer/Real Output - Absolute value of P1 input.

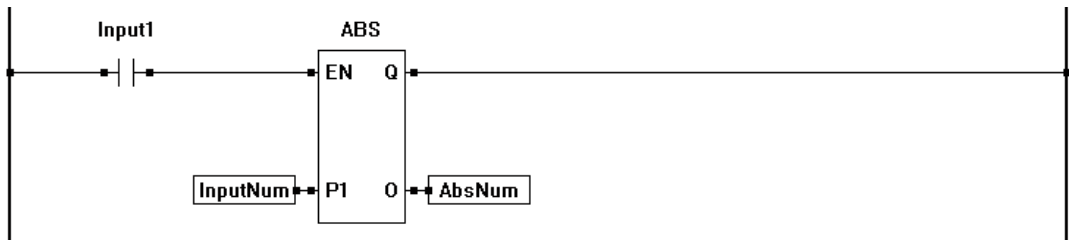
Symbol



Description

The ABS function provides an absolute value output (O) from the input value (P1). The enable (EN) must be true for the ABS function to be enabled. The Q output is true when the ABS function is enabled.

Example Circuit



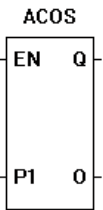
**Related Functions:** ADD, SUB, MULT, DIV

ACOS

Symbol

Inputs

- EN Boolean Function Enable Input - function is disabled if EN is false.
- P1 Real Input - Output is arc cosine of this input.



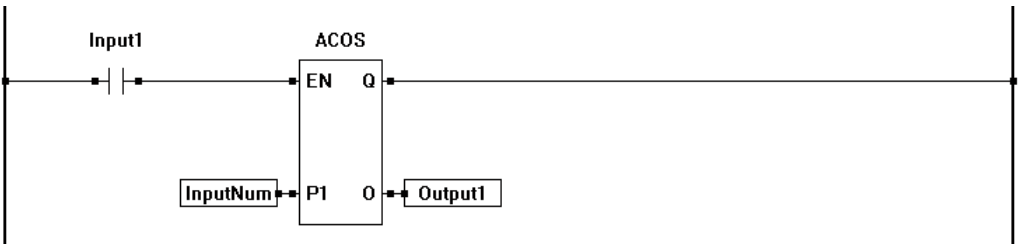
Outputs

- Q Boolean Function Enable Output - true when function is enabled.
- O Real Output - Arc Cosine value of P1 input.

Description

The ACOS function provides an Arc cosine (O) from the input value (P1). The enable (EN) must be true for the ACOS function to be enabled. The Q output is true when the ACOS function is enabled.

Example Circuit



Related Functions: SIN, TAN, ASIN, ATAN, COS

ADD

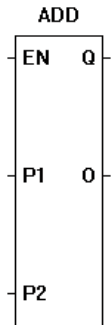
Inputs

- EN Boolean Function Enable Input - function is disabled if EN is false.
- Px Integer/Real Inputs - Output is sum of these inputs.

Outputs

- Q Boolean Function Enable Output - true when function is enabled.
- O Integer/Real Output - Sum of all Px inputs.

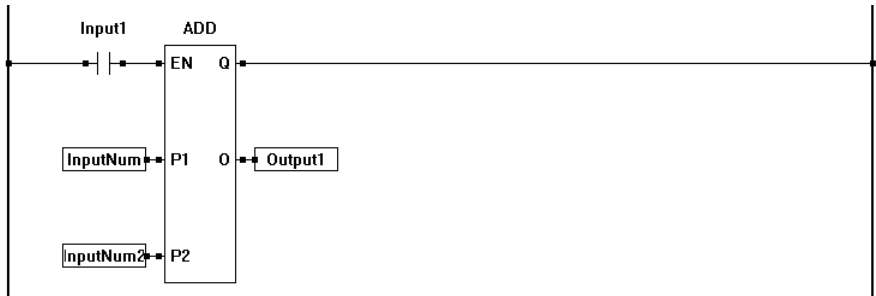
Symbol



Description

The ADD functions sums all the inputs (Px) together and outputs this number (O). The number of inputs is specified when the function is placed in the program. The enable (EN) must be true for the ADD function to be enabled. The Q output is true when the ADD function is enabled.

Example Circuit



Related Functions: SUB, MULT, DIV, ABS



AND

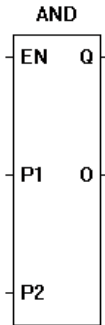
Inputs

- EN Boolean Function Enable Input - function is disabled if EN is false.
- P1 Integer Input - Output is bitwise AND of inputs.
- P2 Integer Input - Output is bitwise AND of inputs.

Outputs

- Q Boolean Function Enable Output - true when function is enabled.
- O Integer Output - Bitwise AND of P1, P2 Inputs.

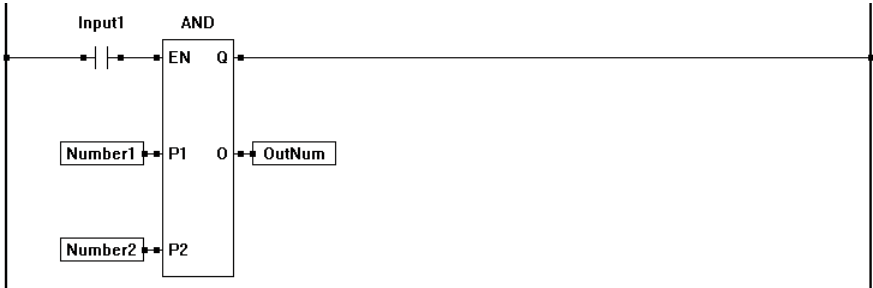
Symbol



Description

The AND functions provides a bitwise AND function of the P1 and P2 inputs. The enable (EN) must be true for the AND function to be enabled. The Q output is true when the AND function is enabled.

Example Circuit



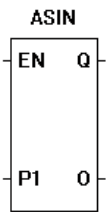
Related Functions: OR, XOR, NOT

ASIN

Inputs

- EN Boolean Function Enable Input - function is disabled if EN is false.
- P1 Real Input - Output is arcsine of this input.

Symbol



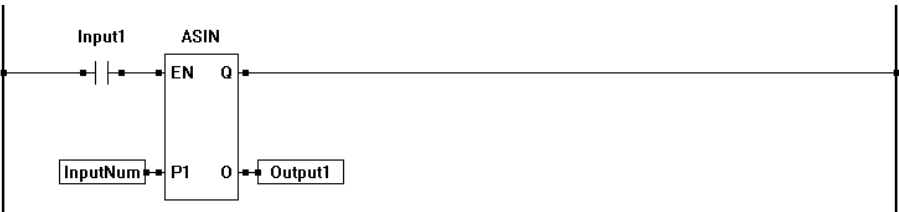
Outputs

- Q Boolean Function Enable Output - true when function is enabled.
- O Real Output - Arcsine value of P1 input.

Description

The ASIN function provides an Arcsine (O) from the input value (P1). The enable (EN) must be true for the ASIN function to be enabled. The Q output is true when the ASIN function is enabled.

Example Circuit



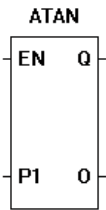
Related Functions: SIN, TAN, ACOS, ATAN, COS

ATAN

Inputs

- EN Boolean Function Enable Input - function is disabled if EN is false.
- P1 Real Input - Output is arctangent of this input.

Symbol



Outputs

- Q Boolean Function Enable Output - true when function is enabled.
- O Real Output - Arctangent value of P1 input.

Description

The ATAN function provides an Arctangent (O) from the input value (P1). The enable (EN) must be true for the ATAN function to be enabled. The Q output is true when the ATAN function is enabled.

Example Circuit



Related Functions: SIN, TAN, ASIN, ACOS, COS

AVG

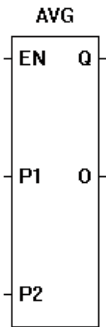
Inputs

- EN Boolean Function Enable Input - function is disabled if EN is false.
- Px Integer/Real Input - Output is average of these inputs.

Outputs

- Q Boolean Function Enable Output - true when function is enabled.
- O Integer/Real Output - Average of all Px inputs.

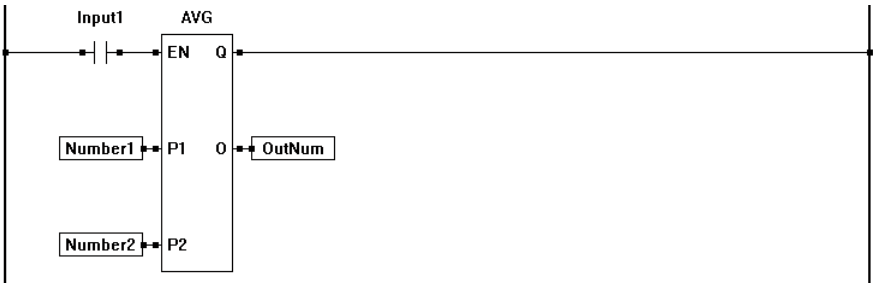
Symbol



Description

The AVG function averages all the inputs (Px) together and outputs this number (O). The number of inputs is specified when the function is placed in the program. The enable (EN) must be true for the AVG function to be enabled. The Q output is true when the AVG function is enabled.

Example Circuit



Related Functions: MAVG, MAX, MIN

BIT\_PACK

Inputs

- EN** Boolean Function Enable Input - function is disabled if EN is false.
- Bx** Boolean Input - The output is a result of these inputs packed.

Outputs

- Q** Boolean Function Enable Output - true when function is enabled.
- O** Integer Output - The integer form of all the boolean input bits.

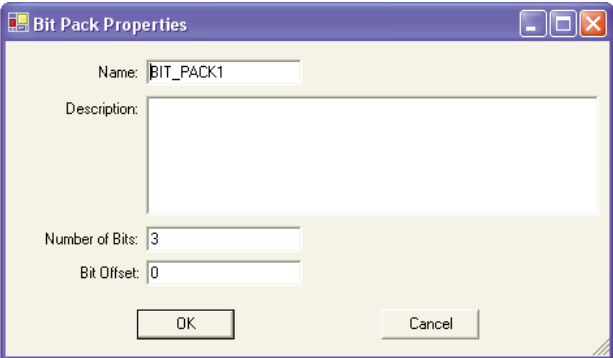
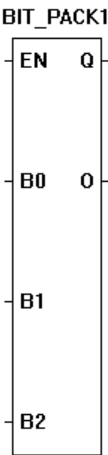
Description

The BIT\_PACK is a configurable function that will convert the inputs bits (from binary) to a single 32-bit integer number. The Bx inputs are the bits to pack, the EN enables the function when true. The Q output is true when the function is enabled. The output O is the 32-bit integer result of the packed inputs.

The *number of bits* must be identified when the function is placed in the ladder diagram (1-32 bits). Only boolean variables or contacts may be used as bit inputs. Included in the configuration is the *bit offset*. The bit offset allows the programmer to use multiple BIT\_PACK functions and have a single 32-bit output integer by offsetting the bit range for each function block. Note the *number of bits + offset bits must be less than or equal to 32*.

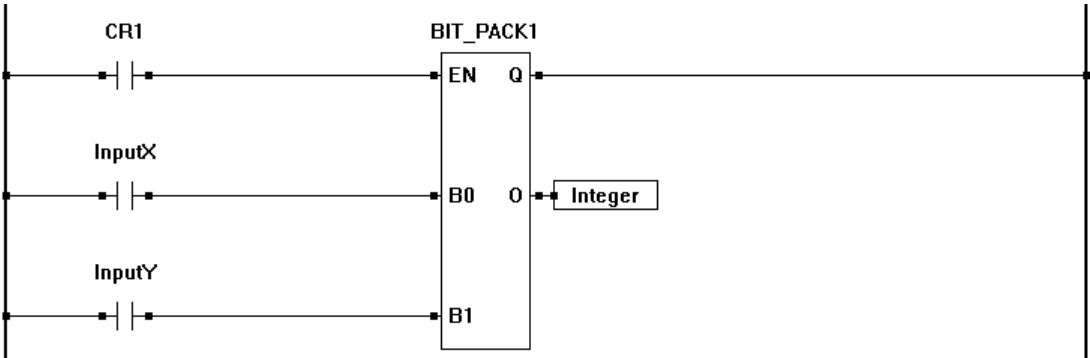
When offsetting (for example, 2 blocks - the first with bits 0-4 and the second with bits 5-10), only the inputs to the particular function will affect the output. When bits change on the second block, the output will change accordingly based on its value 'packed' with the result from the first block's value.

Symbol



BIT True	Integer equivalent
0	1
1	2
2	4
3	8
4	16
5	32
6	64
.	
16	32768
.	
30	1073741824
31	-2147483648

Example Circuit



Related Functions: BIT\_UNPACK

BIT\_UNPACK

Inputs

- EN Boolean Function Enable Input - function is disabled if EN is false.
- I Integer Input - 32-bit integer to be 'unpacked'.

Outputs

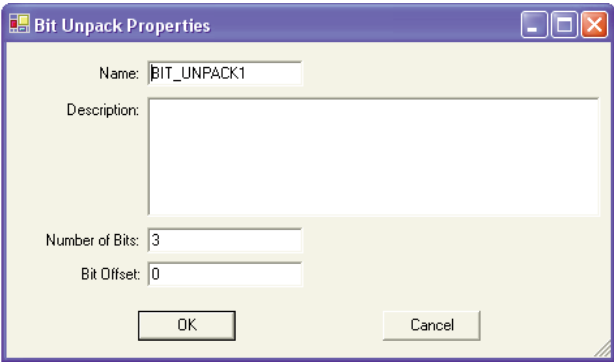
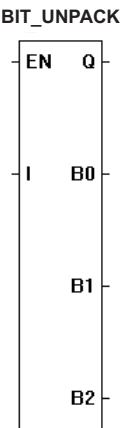
- Q Boolean Function Enable Output - true when function is enabled.
- Bx Boolean Output - The output is a result of these input being converted back to bits from the integer input.

Description

The BIT\_UNPACK is a configurable function that will convert the a 32-bit integer to a 32 individual boolean outputs (bits). The I input is the 32-bit integer input, the EN enables the function when true. The Q output is true when the function is enabled. The Bx outputs are the result of the integer being converted to bit outputs (binary equivalent).

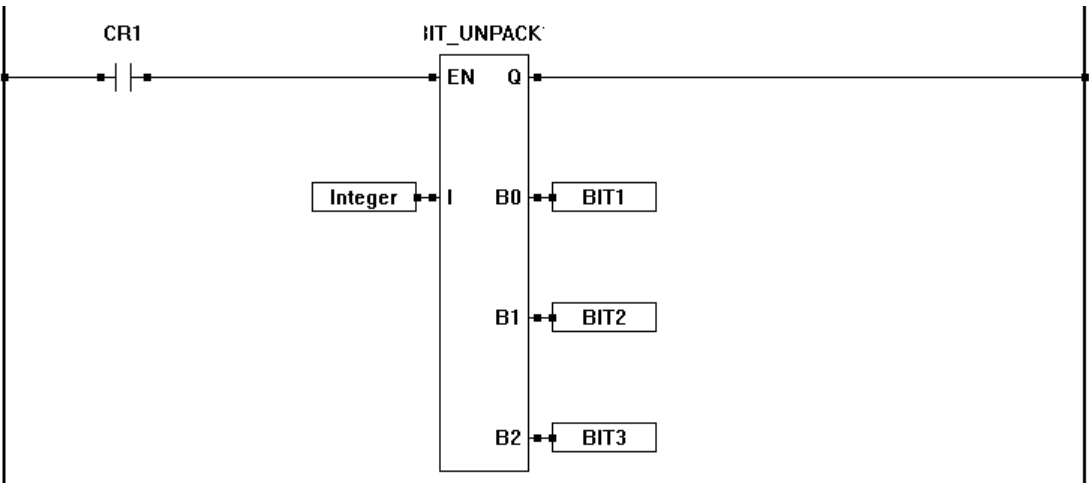
The *number of bits* must be identified when the function is placed in the ladder diagram (1-32 bits). Only boolean variables may be used as bit inputs. Included in the configuration is the *bit offset*. The bit offset allows the programmer to use multiple BIT\_UNPACK functions and have a single 32-bit input integer by offsetting the bit range for each function block. Note the *number of bits + offset bits must be less than or equal to 32*.

Symbol



BIT True	Integer equivalent
0	1
1	2
2	4
3	8
4	16
5	32
6	64
.	
16	32768
.	
30	1073741824
31	-2147483648

Example Circuit



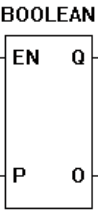
Related Functions: BIT\_PACK

BOOLEAN

Inputs

- EN Boolean Function Enable Input - function is disabled if EN is false.
- P Integer/Real Input - Output is boolean of this input.

Symbol



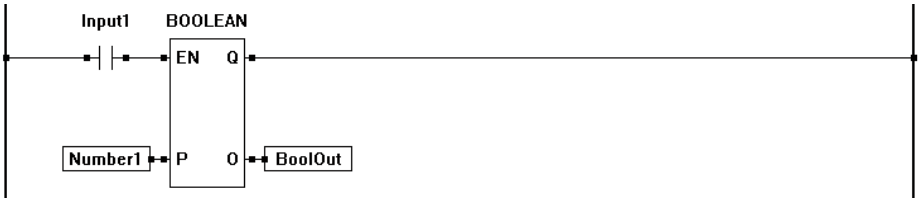
Outputs

- Q Boolean Function Enable Output - true when function is enabled.
- O Boolean output of P input.

Description

The BOOLEAN function converts the input (P) into a boolean (zero or non-zero) output (O). The enable (EN) must be true for the BOOLEAN function to be enabled. The Q output is true when the BOOLEAN function is enabled.

Example Circuit



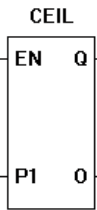
Related Functions: INTEGER, REAL

CEIL

Inputs

- EN Boolean Function Enable Input - function is disabled if EN is false.
- P1 Real Input - Output is the rounded-up result of this input.

Symbol



Outputs

- Q Boolean Function Enable Output - true when function is enabled.
- O Real Output - Rounded-up result of the P1 input.

Description

The CEIL function provides a rounded-up result of the P1 Input and outputs this number (O). The enable (EN) must be true for the CEIL function to be enabled. The Q output is true when the CEIL function is enabled.

Example Circuit



Related Functions: FLOOR



**CMP**

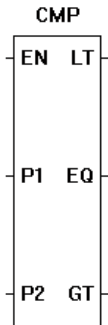
Inputs

- EN** Boolean Function Enable Input - function is disabled if EN is false.
- P1** Integer/Real Input
- P2** Integer/Real Input

Outputs

- LT** Boolean Output - True when  $P1 < P2$ .
- EQ** Boolean Output - True when  $P1 = P2$ .
- GT** Boolean Output - True when  $P1 > P2$ .

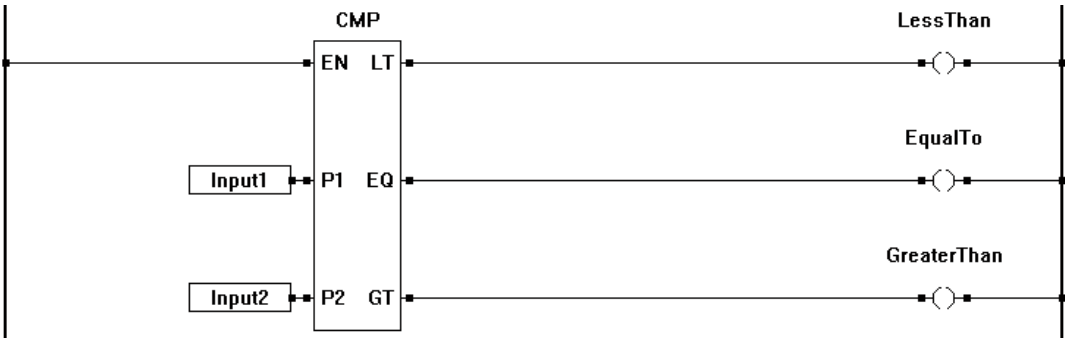
Symbol



Description

The CMP function compares the P1 and P2 inputs. LT is true when the P1 input is less than the P2 input. EQ is true when the P1 input equals the P2 input. GT is true when the P1 input is greater than the P2 input. The enable (EN) must be true for the CMP function to be enabled.

Example Circuit



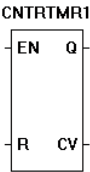
**Related Functions:** LIMIT, HYSTER

CNTRTMR

Inputs

- EN Boolean Input - Enable or Pulse input.
- R Boolean Input - Reset Count

Symbol



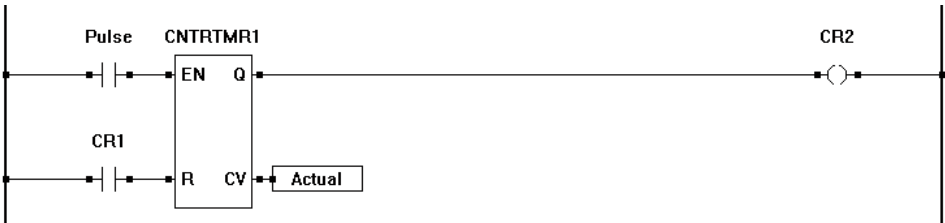
Outputs

- Q Boolean Function Enable Output - true when function is enabled.
- CV Integer Output - Current Counter Value.

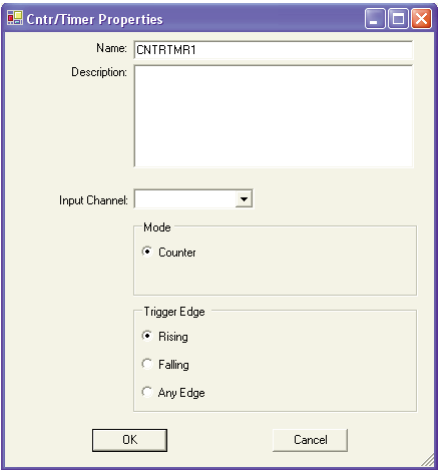
Description

The CNTRTMR function provides access and functionality to a hardware counter (must be supported by target). When configured as a counter, the the trigger can be set to rising or falling edge of the input pulse on the EN input. R resets the current counter value (CV). Q is true when the counter is counting.

Example Circuit



Counter / Timer Properties



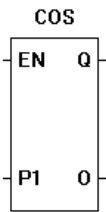
Related Functions: None

COS

Inputs

- EN Boolean Function Enable Input - function is disabled if EN is false.
- P1 Real Input - Output is cosine of this input.

Symbol



Outputs

- Q Boolean Function Enable Output - true when function is enabled.
- O Real Output - Cosine value of P1 input.

Description

The COS function provides a cosine (O) from the input value (P1). The enable (EN) must be true for the COS function to be enabled. The Q output is true when the COS function is enabled.

Example Circuit



Related Functions: SIN, TAN, ASIN, ATAN, ACOS

CTD

Inputs

- CD** Boolean Input - Down Count Pulse.
- LD** Boolean Input - Load Counter Value (sets CV=PV)
- PV** Integer Input - Initial Counter Value

Outputs

- Q** Boolean Function Enable Output - true when CV = 0.
- CV** Integer Output - Current Counter Value.

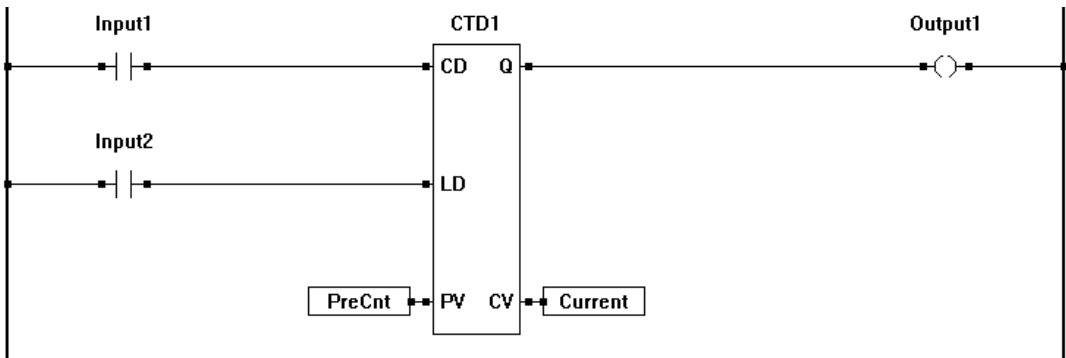
Symbol



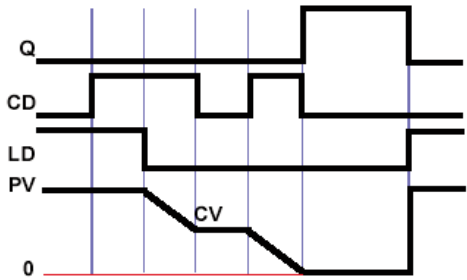
Description

The CTD function is a programmable down counter. A true on CD will cause the counter to decrement by one. Once the counter (CV) equals zero, the Q output will be true. A true on LD will cause the counter to load the PV as the current (CV) count and reset the Q output. The down counter triggers on a false to true transition on the CD input.

Example Circuit



Timing Diagram



Related Functions: CTU, CTUD

CTU

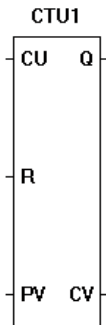
Inputs

- CU** Boolean Input - Up Count Pulse.
- R** Boolean Input - Reset Counter Value (sets CV=0)
- PV** Integer Input - Maximum Counter Value

Outputs

- Q** Boolean Function Enable Output - true when CV = PV.
- CV** Integer Output - Current Counter Value.

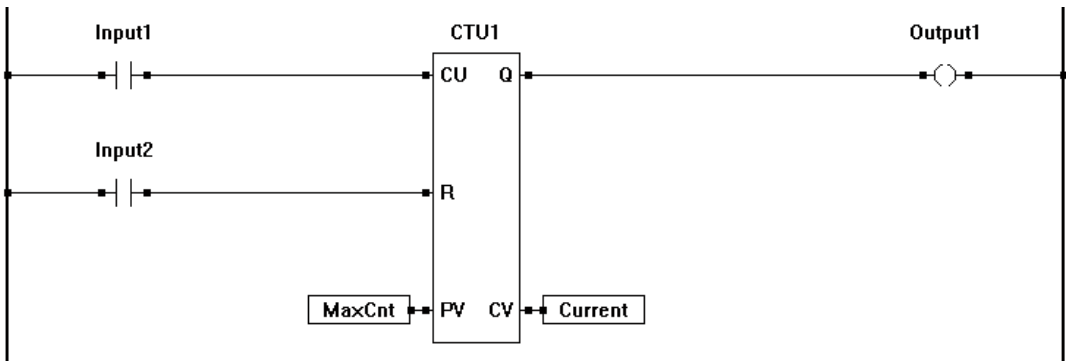
Symbol



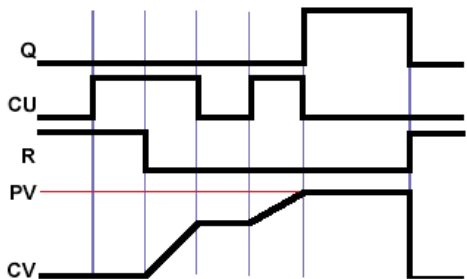
Description

The CTU function is a programmable up counter. A true on CU will cause the counter to increment by one. Once the counter (CV) equals the preset value (CV), the Q output will be true. A true on reset (R) will cause the counter reset to zero and reset the Q output. The down counter triggers on a false to true transition on the CU input.

Example Circuit



Timing Diagram



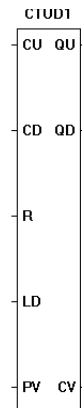
Related Functions:

**CTUD****Inputs**

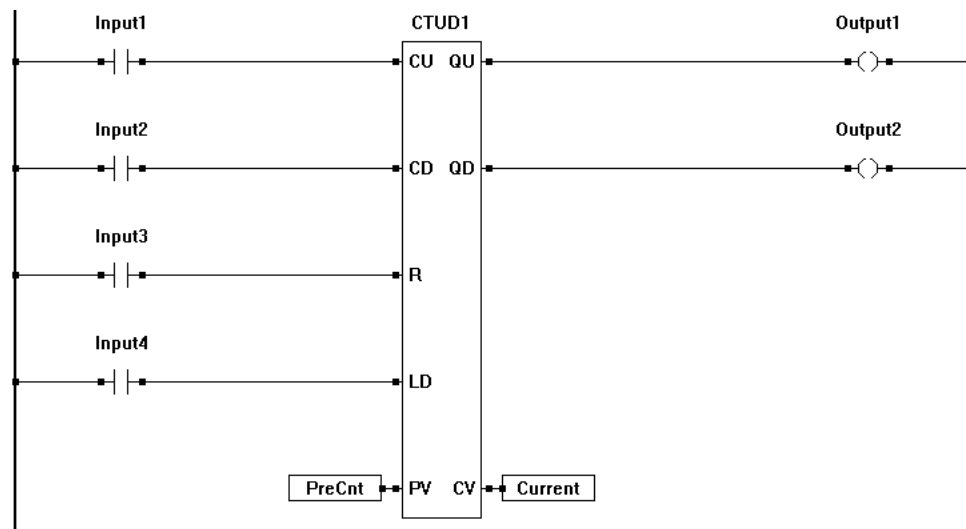
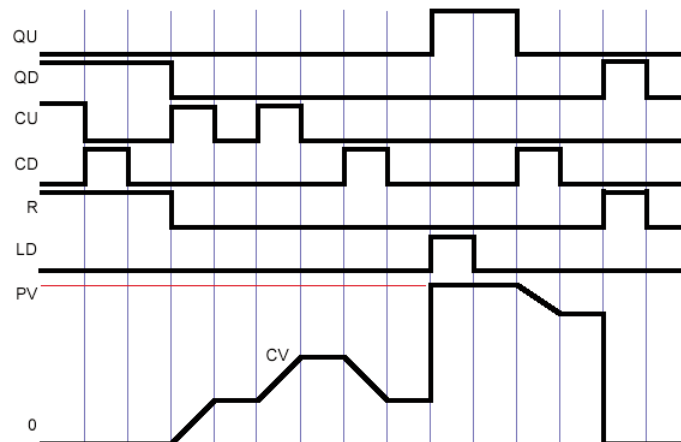
<b>CU</b>	Boolean Input - Up Count Pulse.
<b>CD</b>	Boolean Input - Down Count Pulse.
<b>R</b>	Boolean Input - Reset Counter Value (sets CV=0)
<b>LD</b>	Boolean Input - Load Counter Value (sets CV=PV)
<b>PV</b>	Integer Input - Maximum Counter Value

**Outputs**

<b>QU</b>	Boolean Output - Count up reached, true when CV = PV.
<b>QD</b>	Boolean Output - Zero reached, true when CV = 0.
<b>CV</b>	Integer Output - Current Counter Value.

**Symbol****Description**

The CTUD function is programmable up/down counter. With reset (R) not active, a true on input (CU) will increment the current (CV) count by one, while a true on input (CD) will cause the current count (CV) to decrement by one. When the CV = PV, the output (QU) will be true. When the CV = 0, the output (QD) will be true. A true on the reset (R) will cause CV = 0, QU to go false and QD to go true. A true on the load (LD) will cause CV = PV, QU to go true and QD to go false. The reset (R) is dominant and takes priority over all inputs. The counter inputs trigger on a false to true transition on CU or CD.

**Example Circuit****Timing Diagram**

**Related Functions:** CTD, CTU

---

**DIRECT COIL (Normally De-energized)****Symbol****Inputs**

None

**Outputs**

None

**Description**

The DIRECT COIL is a boolean representation of an internal variable (Control relay) or an actual hardware (real world) output. Its normal state is false or normally de-energized. If there is “power flow” to the DIRECT COIL, then it will be true (on). If there is no “power flow” to the DIRECT COIL, then it will be false (off).

**Example Circuit**

**Related Functions:** INVERTED COIL, LATCH, UNLATCH, DIRECT CONTACT, INVERTED CONTACT

---

**DIRECT CONTACT (Normally Open)**[Symbol](#)[Inputs](#)

None

[Outputs](#)

None

[Description](#)

The DIRECT CONTACT is a boolean representation of an internal variable (Control relay) or an actual hardware (real world) input. Its normal state is false or normally open. A true (on) condition of the input will cause the contact to close and allow “power flow”. A false (off) condition of the input, the contact will be open and not allow “power flow”.

[Example Circuit](#)

**Related Functions:** INVERTED CONTACT, DIRECT COIL, INVERTED COIL, LATCH, UNLATCH



DIV

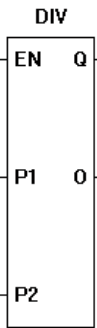
Inputs

- EN Boolean Function Enable Input - function is disabled if EN is false.
- P1 Integer/Real Input - Dividend (# to be divided)
- P2 Integer/Real Input - Divisor (# to divide by)

Outputs

- Q Boolean Function Enable Output - true when function is enabled.
- O Integer/Real Output - result (quotient) of the division.

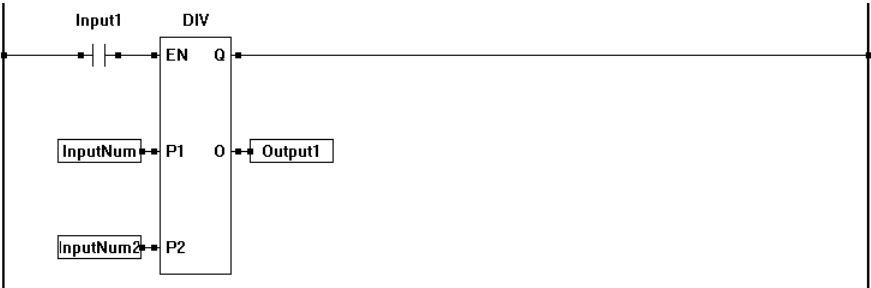
Symbol



Description

The DIV function divides the P1 input by the P2 input and outputs the result (O). The enable (EN) must be true for the DIV function to be enabled. The Q output is true when the DIV function is enabled. The result (O) is the whole number quotient only. No remainder is provided.

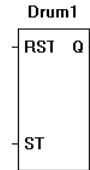
Example Circuit



Related Functions: ADD, SUB, MULT, ABS

**DRUM\_SEQ****Symbol****Inputs**

- RST** Boolean Function Reset Input - function is disabled if RST is true.  
**ST** Boolean Input - Step Input-Causes increment of sequencer step.

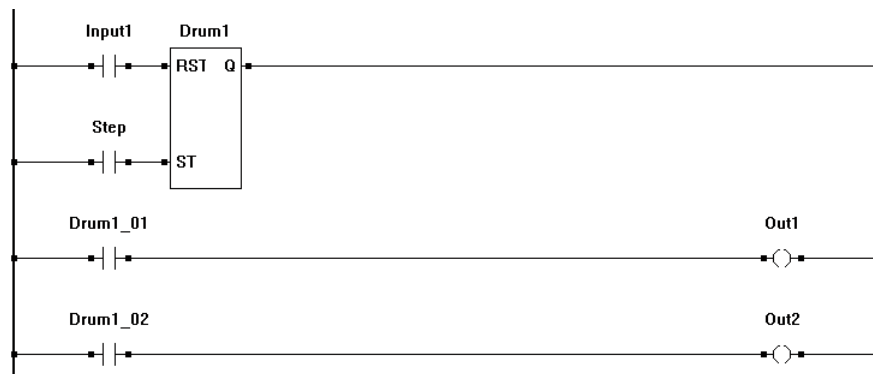
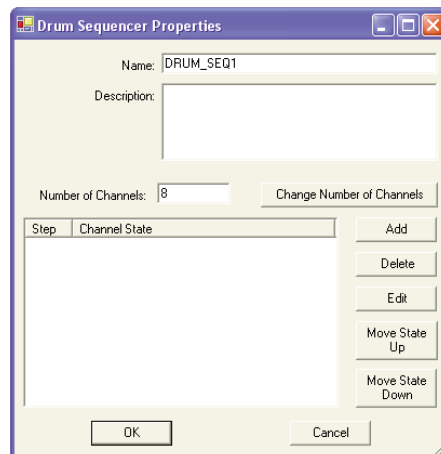
**Outputs**

- Q** Boolean Function Enable Output - true when function is enabled.

**Description**

The DRUM\_SEQ function steps through a pre-programmed set of on/off channel (contact) settings when the ST input is pulsed. The drum sequence is programmed with a maximum of 32 channels. Each channel may be programmed to be on or off (1 or 0) for each step. When the ST input is pulsed, the drum sequencer increments to the next step and the channel's contacts change to the appropriate state.

Each channel is designed to represent a contact. When the channel is set to 0 (off), its contact is false. When the channel is set to 1 (on), its contact is true. The contacts are automatically named according to the DRUM\_SEQ name. The channel settings are defined in a matrix. For each step in the drum sequencer, using the check box, select the state of each channel for each drum sequencer step.

**Example Circuit****Channel / Step Example**

**Related Functions:** None

EQUAL TO (=)

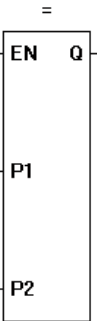
Inputs

- EN Boolean Function Enable Input - function is disabled if EN is false.
- Px Integer/Real Input - Output is true if these inputs are equal.

Outputs

- Q Boolean Function Enable Output - true when all inputs are equal..

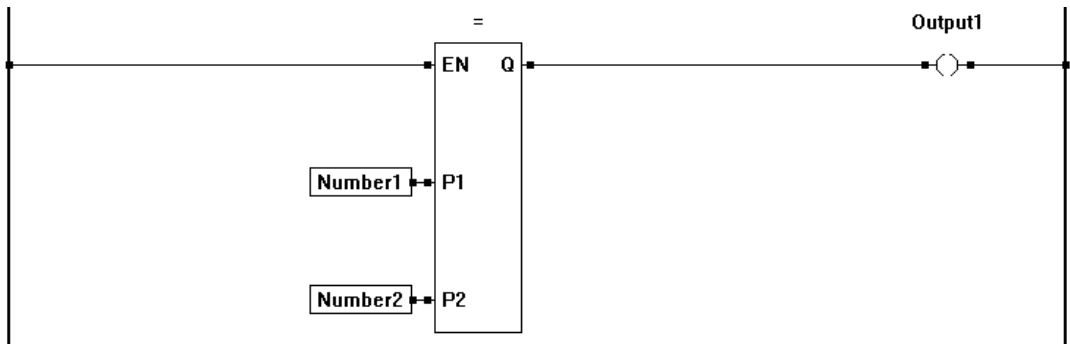
Symbol



Description

The EQUAL TO function provides an equal to comparison for the Px inputs. The number of inputs is specified when the object is placed. The Q output is true if all the Px inputs are equal. The Enable must be true for the EQUAL TO function to be enabled.

Example Circuit



Related Functions: <, >, <=, >=, <>

EEPROM\_READ

Inputs

- EN Boolean Function Enable Input - function is disabled if EN is false.
- AD Integer Input - Address to read value from (0-2906)-See Below

Outputs

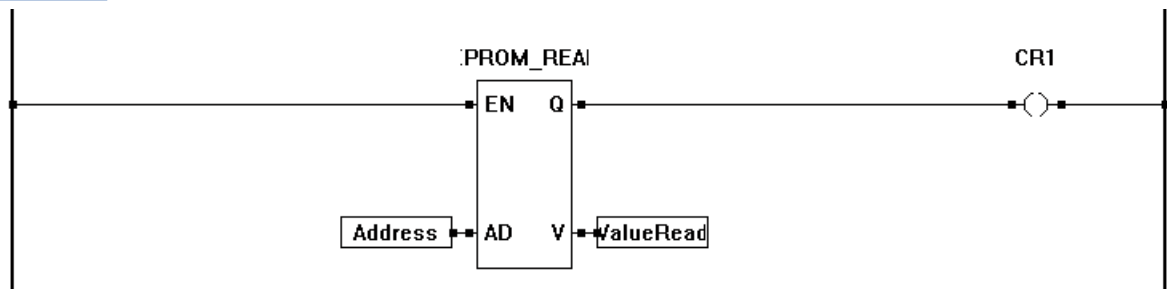
- Q Boolean Function Enable Output - true when has completed the EEPROM read cycle.
- V Integer, Real, Timer or Boolean Output - Actual value read from EEPROM

Description

The EEPROM\_READ recalls variables stored in non-volatile memory (EEPROM). The function is enabled when EN is true. AD provides the actual address to read from and V is the actual value that is read from the EEPROM. Q is true when the read cycle has completed. Note: The same variable type that writes to the EEPROM location should be used to read the EEPROM location.

EEPROM\_READ is available only on 256K PLC on a Chip target models.

Example Circuit



Other

Each EEPROM address is absolute and is one byte in size. Boolean variables fill two bytes while all other variable types fill four bytes of EEPROM. When reading variables from EEPROM storage, it is important that use the exact address location for the variable only (taking into account variable types and sizes). A memory map is recommended for organizing variables stored in EEPROM.

See EEPROM\_WRITE for more on how variables are written to EEPROM storage.

Related Functions:

EEPROM\_WRITE

Inputs

- EN Boolean Function Enable Input - function is disabled if EN is false.
- AD Integer Input - Address to write value to (0-2906)-See Below
- V Integer, Real, Timer, or Boolean Input - Actual value to write to EEPROM address.

Outputs

- Q Boolean Function Enable Output - true when has completed the EEPROM write cycle without error.

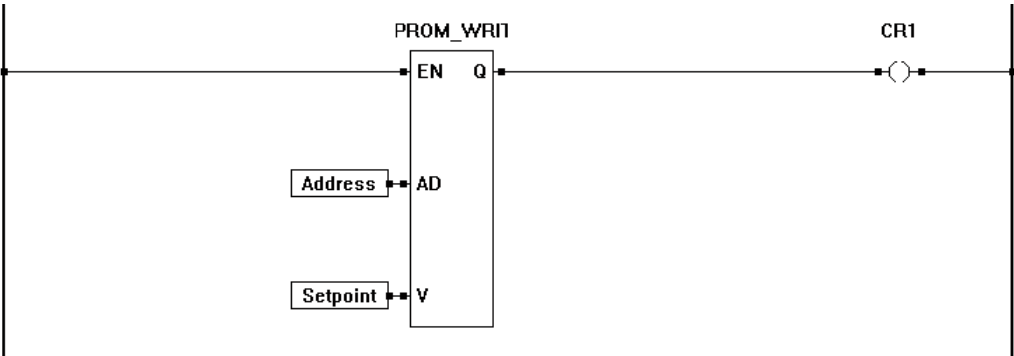
Description

The EEPROM\_WRITE function allows variables (integer) to be stored in non-volatile memory (EEPROM). The function is enabled when EN is true. AD provides the actual address to write to EEPROM and V is the actual value that is written. The write occurs on the low to high transition of EN. Q is true when the write cycle has completed without error.

EEPROM\_READ is available only on 256K PLC on a Chip target models. The same variable type that writes to the EEPROM location should be used to read the EEPROM location.

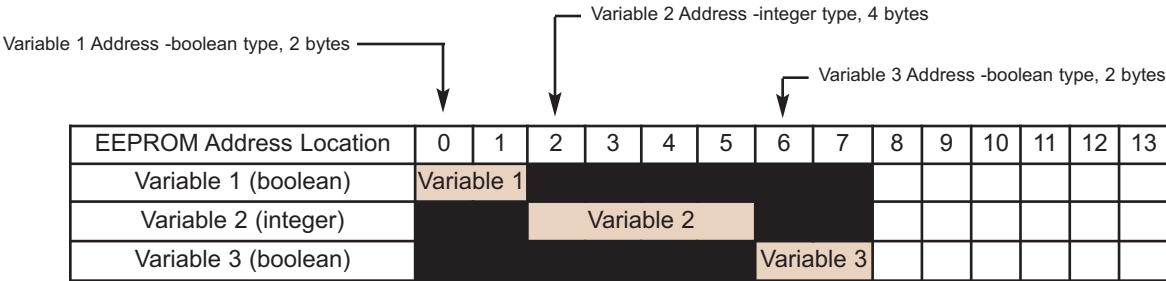
Writing to EEPROM is a relatively slow operation and must be considered while writing the program (scan time). EEPROM storage area has a limited number of write cycles; therefore it shouldn't be used to store data which changes often and must be re-written often.

Example Circuit



Other

Each EEPROM address is absolute and is one byte in size. Boolean variables fill two bytes while all other variable types fill four bytes of EEPROM. When writing a boolean to address 0, the actual variable will use addresses 0 and 1 (two bytes). Should you write an integer variable into address 0, then it would use addresses 0-3. A memory map should be created and used to assign variable types and addresses prior to coding to ensure that variable size and types are accounted for.



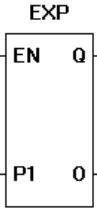
Related Functions: EEPROM\_READ

EXP

Inputs

- EN Boolean Function Enable Input - function is disabled if EN is false.
- P1 Real Input - Output is natural exponential of this number.

Symbol



Outputs

- Q Boolean Function Enable Output - true when function is enabled.
- O Real Output - natural exponential of the P1 input.

Description

The EXP function provides the natural exponential of the P1 input. The output (O) is the natural exponential of the P1 input. The enable (EN) must be true for the EXP function to be enabled.

Example Circuit



Related Functions: EXPT, SQRT, LN, MOD, LOG

EXPT

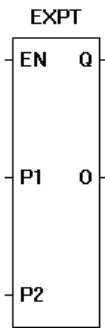
Inputs

- EN Boolean Function Enable Input - function is disabled if EN is false.
- P1 Real Input - Base number
- P2 Real Input - Exponent number

Outputs

- Q Boolean Function Enable Output - true when function is enabled.
- O Real Output - Result of exponentiation.

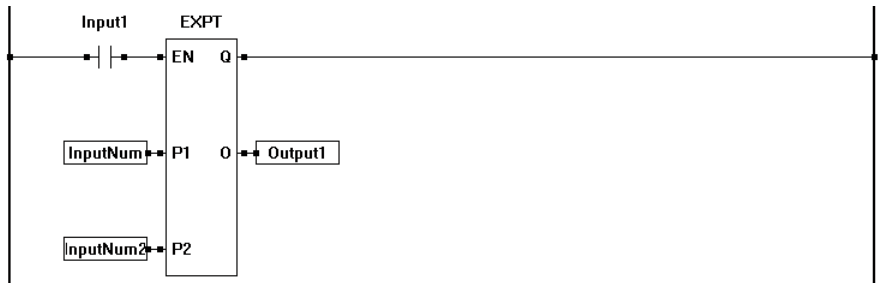
Symbol



Description

The EXPT function provides the exponentiation of the P1 and P2 inputs. The output (O) is the result of the exponentiation ( $P1^{P2}$ ). The enable (EN) must be true for the EXPT function to be enabled.

Example Circuit



Related Functions: EXP, SQRT, LN, MOD, LOG

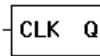
F\_TRIG

Inputs

CLK Boolean Function Enable Input - detects falling edge of CLK.

Symbol

F\_TRIG1



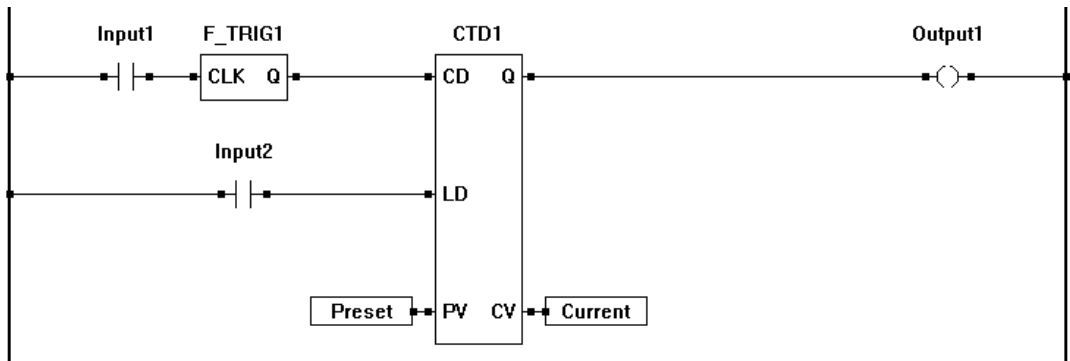
Outputs

Q Boolean Output - Pulsed Output, true for one scan when CLK detects a falling edge.

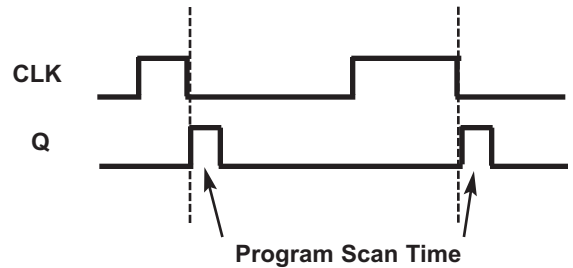
Description

The F\_TRIG is a function that may be used to trigger another function on the falling edge of a transition. When the CLK detects a true to false transition, the output (Q) is energized (for one scan of the program only).

Example Circuit



Timing Diagram



Related Functions: R\_TRIG

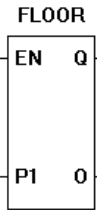


FLOOR

Inputs

- EN Boolean Function Enable Input - function is disabled if EN is false.
- P1 Real Input - Output is rounded-down result of this input number.

Symbol



Outputs

- Q Boolean Function Enable Output - true when function is enabled.
- O Real Output - Rounded-down result of input.

Description

The FLOOR function provides a rounded-down output of P1 input. The output (O) is the is the rounded-down number. The enable (EN) must be true for the FLOOR function to be enabled.

Example Circuit



Related Functions: CEIL

## GC\_SSI

### Symbol

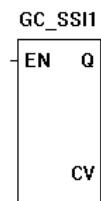
#### Inputs

**EN** Boolean Function Enable Input - function is disabled if EN is false.

#### Outputs

**Q** Boolean Function Enable Output - true when function is enabled and there is communication to the encoder. False when there is a communication error.

**CV** Integer representation of the Encoder's output.



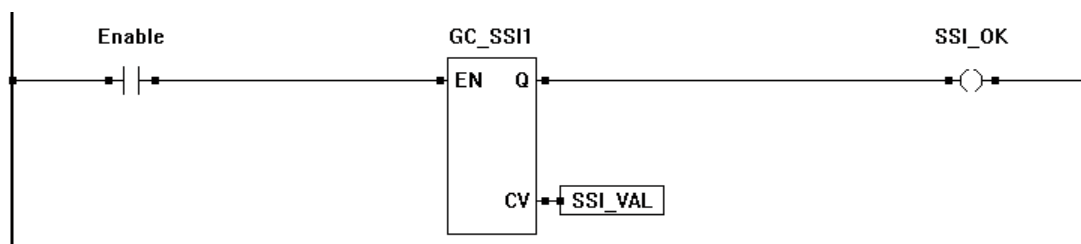
### Description - Single or Master GC\_SSI

The GC\_SSI function is used to interface to encoders that support Gray Code, Synchronous Serial Interface. The interface is via the PLC on a Chip or target's SPI interface port. This function is not available on targets that do not support it. The target must be configured properly (if supported) to allow the GC\_SSI function to be selected and placed.

The GC\_SSI communicates serially over the SPI port to the encoder (additional interface circuitry required.) The Output is an Integer representation of the encoder's value. The encoder value is read (graycode) and then converted into a binary number. This number is represented as an integer output.

The GC\_SSI Block must be configured to match the encoder's and cable specifications.

### Example Circuit



### Configuration Details

#### Single / Master GC\_SSI Setup

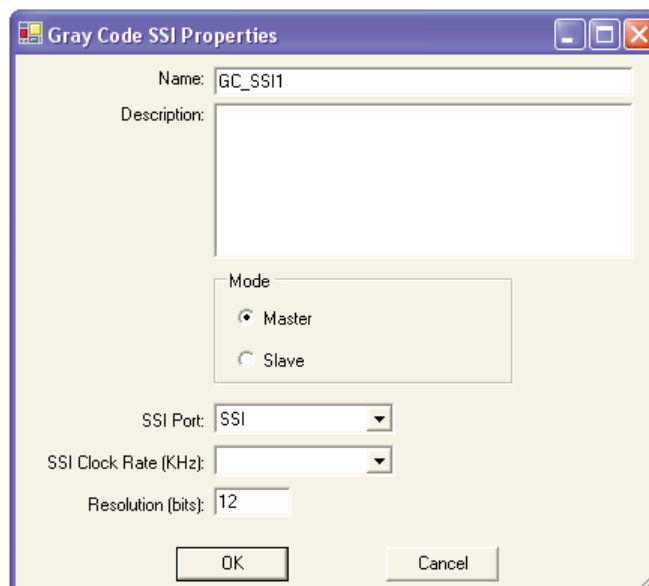
**Name:** Name of the function.

**Description:** Description of what function is used for.

**SSI Port:** Select the SSI Port to use.

**SSI Clock Rate:** Select the clock rate / baud rate for the encoder communication. This is dependent on the encoder and cable length. Refer the encoder's documentation for this setting.

**Resolution:** Resolution of the Encoder. This is dependent on the encoder. Refer the encoder's documentation for this setting.



## GC\_SSI (Continued)

### Description - Slave GC\_SSI

The GC\_SSI function may be configured as a slave (target dependent). When configured as a slave (must be connected with another target acting as the master), the slave SSI port will receive the same data that the master does from an SSI encoder. This provides the option of “connecting” two or more targets to a single SSI encoder to provide redundancy.

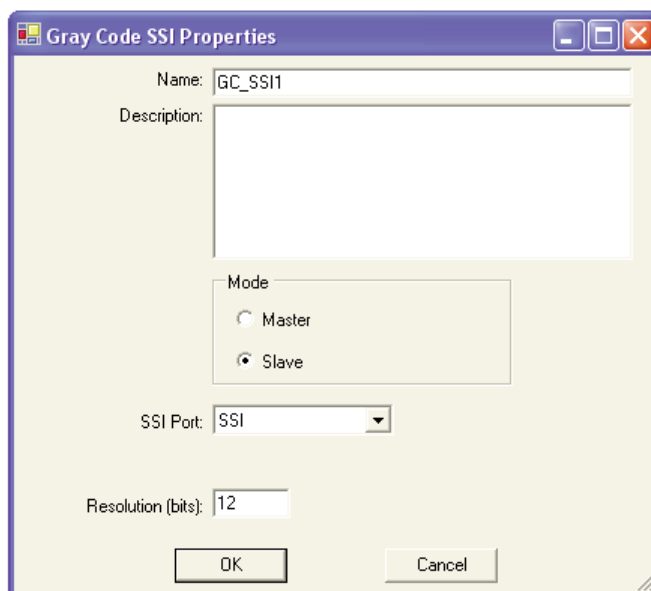
When configuring as a slave, you must select **Slave, SSI Port** and enter the **Resolution**. The Clock rate is determined by the *Master* SSI.

Note: The resolution must match the resolution programmed on the *Master* and the resolution of the SSI encoder.

### Configuration Details

#### Slave GC\_SSI Setup

Name:	Name of the function.
Description:	Description of what function is used for.
SSI Port:	Select the SSI Port to use.
Resolution:	Resolution of the Encoder. This is dependent on the encoder. Refer the encoder's documentation for this setting.



Gray Code SSI Properties

Name: GC\_SSI1

Description:

Mode

☐ Master

☒ Slave

SSI Port: SSI

Resolution (bits): 12

OK Cancel

**Related Functions:** None

GETDATE

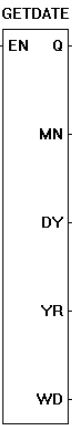
Inputs

**EN** Boolean Function Enable Input - function is disabled if EN is false.

Outputs

**Q** Boolean Function Enable Output - true when function is enabled.  
**MN** Integer Output - Month value (1-12)  
**DY** Integer Output - Day value (1-31).  
**YR** Integer Output - Year value (0-99).  
**WD** Integer Output - Day of the Week value (1-7).

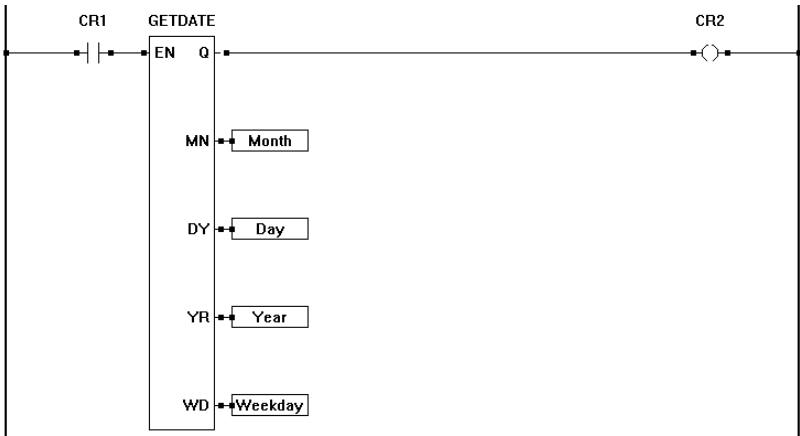
Symbol



Description

The GETDATE function reads the current date from the hardware real time clock. The values of the date are stored into the integer variables on the outputs. The enable (EN) must be true for the GETDATE function to be enabled.

Example Circuit



**Related Functions:** GETTIME, SETDATE, SETTIME

GETTIME

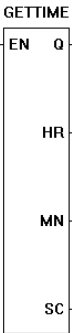
Inputs

**EN** Boolean Function Enable Input - function is disabled if EN is false.

Outputs

**Q** Boolean Function Enable Output - true when function is enabled.  
**HR** Integer Output - Hours value (0-23).  
**MN** Integer Output - Minutes value (0-59).  
**SC** Integer Output - Seconds value (0-59).

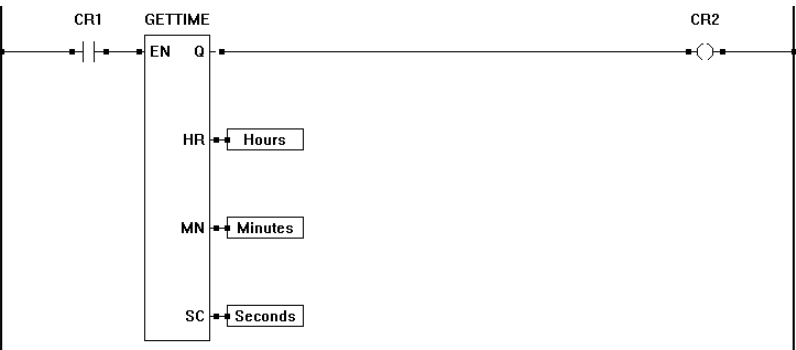
Symbol



Description

The GETTIME function reads the current time from the hardware real time clock. The values of the time are stored into the integer variables on the outputs. The enable (EN) must be true for the GETTIME function to be enabled.

Example Circuit



**Related Functions:** SETTIME, GETDATE, SETDATE

GREATER THAN (>)

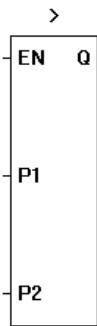
Inputs

- EN Boolean Function Enable Input - function is disabled if EN is false.
- Px Integer/Real Input - Input numbers for comparison.

Outputs

- Q Boolean Output - true when greater than conditions are met.

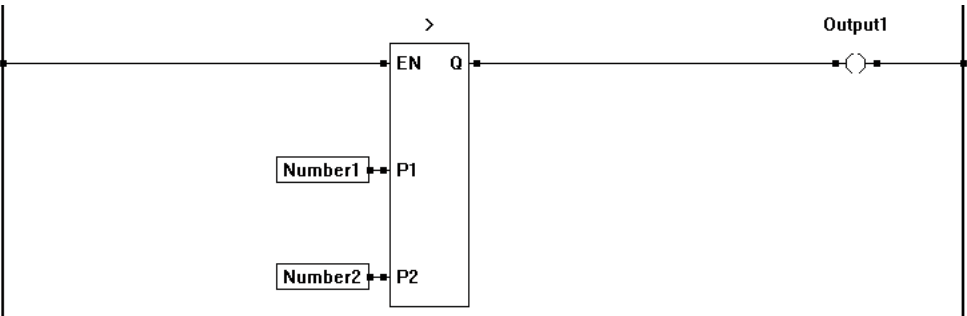
Symbol



Description

The GREATER THAN provides an if greater than comparison for the Px inputs. The number of inputs is specified when the object is placed. The output (Q) is true if P1 is greater than P2 and P2 is greater than P3 and so on. The enable (EN) must be true for the GREATER THAN function to be enabled.

Example Circuit



Related Functions: <, =, <=, >= <>

GREATER THAN OR EQUAL TO (>=)

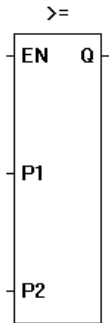
Inputs

- EN Boolean Function Enable Input - function is disabled if EN is false.
- Px Integer/Real Input - Input numbers for comparison.

Outputs

- Q Boolean Output - true when greater than or equal to conditions are met.

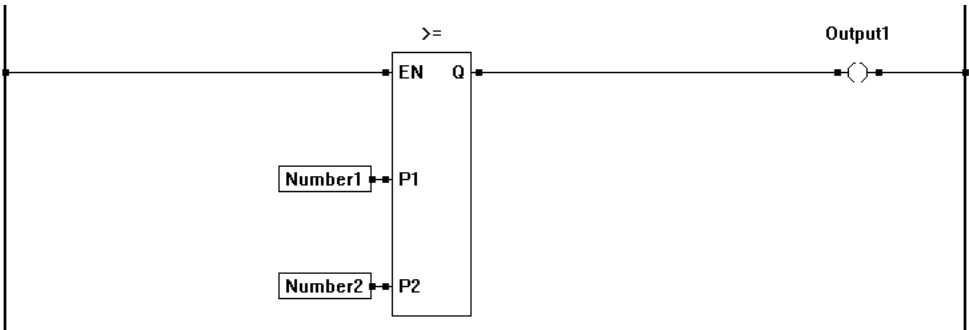
Symbol



Description

The GREATER THAN OR EQUAL TO provides an if greater than or equal to comparison for the Px inputs. The number of inputs is specified when the object is placed. The output (Q) is true if P1 is greater than or equal to P2 and P2 is greater than or equal to P3 and so on. The enable (EN) must be true for the GREATER THAN OR EQUAL TO function to be enabled.

Example Circuit



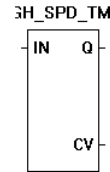
Related Functions: <, =, >, <=, >=

HIGH\_SPD\_TMR

Symbol

Inputs

IN      Boolean Function Enable Input - Rising Edge Triggered



Outputs

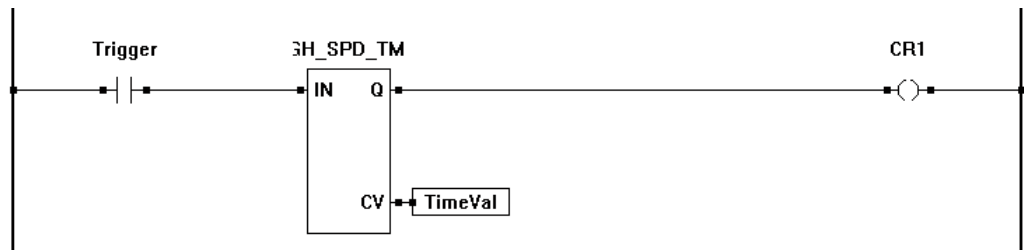
Q      Boolean Output - true when IN is true.  
CV      Integer - Timer output value in 100 microsecond increments.

Description

The HIGH\_SPD\_TMR is a 100 microsecond resolution timer. When the IN detects a rising edge transition, the timer resets and begins timing from zero. When the IN detects a falling edge transition, the timer latches the current timer value. CV holds the current elapsed time when the timer is timing and the latched elapsed timer value when the timer stops timing. The output will be in 100 microsecond increments as an integer.

Example: if the CV is 1000 then the actual time would be 100 milliseconds.

Example Circuit



Related Functions:



HYSTER

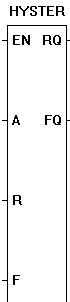
Inputs

- EN Boolean Function Enable Input - function is disabled if EN is false.
- A Real Input - Actual input.
- R Real Input - Rise input.
- F Real Input - Fall input.

Outputs

- RQ Boolean Output - true when actual (A) is > rise (R).
- FQ Boolean Output - true when actual (A) is < fall (F).

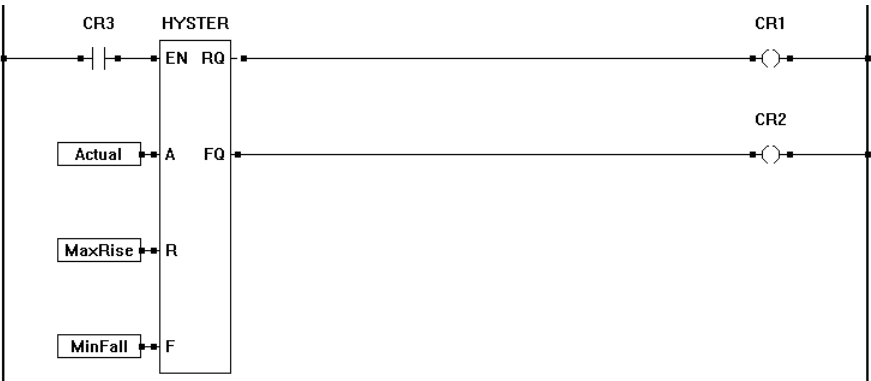
Symbol



Description

The HYSTER provides hysteresis into a control loop. When the actual (A) is greater than the rise (R), then output RQ is true and FQ is false. When actual (A) is less than fall (F), the output FQ is true and RQ is false. The enable (EN) must be true for the HYSTER function to be enabled.

Example Circuit



Related Functions: LIMIT

INTEGER

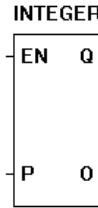
Inputs

- EN Boolean Function Enable Input - function is disabled if EN is false.
- P Boolean/Real /Timer Input - Output is integer of this input.

Outputs

- Q Boolean Function Enable Output - true when function is enabled.
- O Integer Output - Integer conversion of P input. If the input type is a timer, then the output is a integer representation in milliseconds of the input value.

Symbol



Description

The INTEGER function converts the input (P) into an integer ouput (O). The enable (EN) must be true for the INTEGER function to be enabled. The Q output is true when the INTEGER function is enabled.

Example Circuit



Related Functions: REAL, BOOL

## INVERTED COIL

### Symbol

### Inputs

None



### Outputs

None

### Description

The INVERTED COIL is a boolean representation of an internal variable (Control relay) or an actual hardware (real world) output. Its normal state is true or normally energized. If there is “power flow” to the DIRECT COIL, then it will be false (off). If there is no “power flow” to the DIRECT COIL, then it will be true (on).

### Example Circuit



**Related Functions:** DIRECT COIL, LATCH, UNLATCH

---

## INVERTED CONTACT

### Symbol

### Inputs

None



### Outputs

None

### Description

The INVERTED CONTACT is a boolean representation of an internal variable (Control relay) or an actual hardware (real world) input. Its normal state is true or normally closed. A true (on) condition of the input will cause the contact to open and stop “power flow”. A false (off) condition of the input, the contact will be closed and allow “power flow”.

### Example Circuit



**Related Functions:** INVERTED CONTACT

J1939\_SPN

Inputs

EN      Boolean Function Enable Input - function is disabled if EN is false.

Outputs

Q      Boolean Function Enable Output - true when j1939 data is valid.  
ERR    Integer Output - Error output. See Error code table.  
VAL    Integer / Real Value of actual SPN data in engineering units.

Symbol

J1939\_SPN

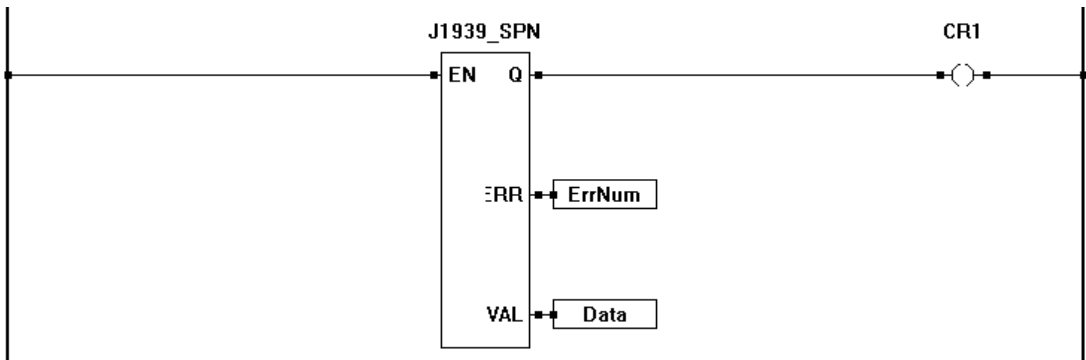


Description

When placing the J1939\_SPN function, the actual SPN (Suspect Parameter Number) must be selected (See SPN Listing).

When enable is true, the function is active. The Q output is true only when the J1939 data for the specified SPN is valid (false when not valid). The ERR output is an integer number representing error codes that correspond to communication issues with the selected SPN and the J1939 bus. The VAL output is the actual value of the parameter that was received (on the selected SPN). This value is in engineering units (based on how the target is configured).

Example Circuit



Error Codes

The error codes for the J1939\_SPN block are:

<u>Code Number</u>	<u>Title</u>	<u>Description</u>
-1	SPN NOT FOUND	The SPN number is not a currently supported number on the PLC on a Chip target controller.
-2	PGN NOT FOUND	The PGN number is not a currently supported number on the PLC on a Chip target controller.
-3	VALUE NOT AVAILABLE	The data value for the specified SPN was not available. Possible cause may be the engine does not support this parameter.
-4	VALUE ERROR / RESERVED	The was an error retrieving the SPN value or the SPN is reserved. Possible cause may be a sensor is malfunctioning and a value cannot be read.

## J1939\_SPN (Con't)

Supported Suspect Parameter Numbers (SPN)

SPN	Parameter	PGN	Bit Size	Bit Start	Res Gain	Res Offset	Metric Units	English Units
22	Extended Crankcase Blow-by Pressure	65263	8	8	0.05	0	kPa	psi
51	Throttle Position	65266	8	48	0.4	0	%	%
52	Engine Intercooler Temperature	65262	8	48	1	-40	C	F
81	Particulate Trap Inlet Pressure	65270	8	0	0.5	0	kpA	psi
84	Wheel-based Vehicle Speed	65265	16	8	0.00391	0	km/h	mph
86	Cruise Control Set Speed	65265	8	40	1	0	km/h	mph
91	Accelerator Pedal(AP) Position	61443	8	8	0.4	0	%	%
92	Percent Load at Current Speed	61443	8	16	1	0	%	%
94	Fuel Delivery Pressure	65263	8	0	4	0	kPa	psi
98	Engine Oil Level	65263	8	16	0.4	0	%	%
100	Engine Oil Pressure	65263	8	24	4	0	kPa	psi
101	Crankcase Pressure	65263	16	32	0.00781	-250	kPa	psi
102	Boost Pressure	65270	8	8	2	0	kpA	psi
105	Intake Manifold 1 Temperature	65270	8	16	1	-40	C	F
106	Air Inlet Pressure	65270	8	24	2	0	kpA	psi
107	Air Filter Differential Pressure	65270	8	32	0.05	0	kpA	psi
109	Coolant Pressure	65263	8	48	2	0	kPa	psi
110	Engine Coolant Temperature	65262	8	0	1	-40	C	F
111	Coolant Level	65263	8	56	4	0	kPa	psi
112	Coolant Filter Differential Pressure	65270	8	56	0.5	0	kpA	psi
114	Net Battery Current	65271	8	0	1	-125	A	A
115	Alternator Current	65271	8	8	1	0	A	A
123	Clutch Pressure	65272	8	0	16	0	kPa	psi
124	Transmission Oil Level	65272	8	8	0.4	0	%	%
126	Transmission Filter Differential Pressure	65272	8	16	2	0	kPa	psi
127	Transmission Oil Pressure	65272	8	24	16	0	kPa	psi
158	Battery Potential	65271	16	48	0.05	0	V	V
167	Alternator Potential	65271	16	16	0.05	0	V	V
168	Electrical Potential	65271	16	32	0.05	0	V	V
173	Exhaust Gas Temperature	65270	16	40	0.03125	-273	C	F
174	Fuel Temperature	65262	8	8	1	-40	C	F
175	Engine Oil Temperature 1	65262	16	16	0.03125	-273	C	F
176	Turbo Oil Temperature	65262	16	32	0.03125	-273	C	F
177	Transmission Oil Temperature	65272	16	32	0.03125	-273	C	F
183	Fuel Rate	65266	16	0	0.05	0	L/h	G/h
184	Instantaneous Fuel Economy	65266	16	16	0.00195	0	km/L	mpg
185	Average Fuel Economy	65266	16	32	0.00195	0	km/L	mpg
188	Engine Speed At Idle, Point 1	65251	16	0	0.125	0	rpm	rpm
190	Engine Speed	61444	16	24	0.125	0	rpm	rpm

SPNs continued on next page

**J1939\_SPN (Con't)**

SPN	Parameter	PGN	Bit Size	Bit Start	Res Gain	Res Offset	Metric Units	English Units
512	Driver's Demand Torque	61444	8	8	1	-125	%	%
513	Actual Engine Torque	61444	8	16	1	-125	%	%
523	Current Gear	61445	8	24	1	-125		
524	Selected Gear	61445	8	0	1	-125		
526	Actual Gear Ratio	61445	16	8	0.001	0		
528	Engine Speed At Point 2	65251	16	24	0.125	0	rpm	rpm
529	Engine Speed At Point 3	65251	16	48	0.125	0	rpm	rpm
530	Engine Speed At Point 4	65251	16	72	0.125	0	rpm	rpm
531	Engine Speed At Point 5	65251	16	96	0.125	0	rpm	rpm
532	Engine Speed At High Idle, Point 6	65251	16	120	0.125	0	rpm	rpm
533	Maximum Momentary Engine Override Speed, Point 7	65251	16	168	0.125	0	rpm	rpm
534	Maximum Momentary Engine Override Time Limit	65251	8	184	0.1	0	s	s
535	Requested Speed Control Range Lower Limit	65251	8	192	10	0	rpm	rpm
536	Requested Speed Control Range Upper Limit	65251	8	200	10	0	rpm	rpm
537	Requested Speed Control Torque Lower Limit	65251	8	208	1	-125	%	%
538	Requested Speed Control Torque Upper Limit	65251	8	216	1	-125	%	%
539	Percent Torque At Idle, Point 1	65251	8	16	1	-125	%	%
540	Percent Torque At Point 2	65251	8	40	1	-125	%	%
541	Percent Torque At Point 3	65251	8	64	1	-125	%	%
542	Percent Torque At Point 4	65251	8	88	1	-125	%	%
543	Percent Torque At Point 5	65251	8	112	1	-125	%	%
544	Reference Engine Torque	65251	16	152	1	0	Nm	lb-ft
545	Gain (KP) of Endspped Governor	65251	16	136	0.00078	0	%/rpm	%/rpm
974	Remote Accelerator	61443	8	24	0.4	0	%	%
1134	Engine Intercooler Thermostat Opening	65262	8	56	0.4	0	%	%

**Parameter Group Number Info (PGN)**

PGN	Description	Abbrev	Repetition Rate	Default Priority	SPNs	Data Length
61443	Electronic Engine Controller 2	EEC2	50 ms	3	91 92 558 559 974 1437	8
61444	Electronic Engine Controller 1	EEC1	20 ms	3	190 512 513 899 1483 1675 2432	8
61445	Electronic Transmission Controller 2	ETC2	100 ms	6	162 163 523 524 526	8
65251	Engine Configuration	EC	5 s	6	1 542 543 544 545 1712 1794 1846	34
65262	Engine Temperature 1	ET1	1 s	6	52 110 174 175 176 1134	8
65263	Engine Fluid Level/Pressure 1	EEFL/P1	500 ms	6	22 94 98 100 101 109 111	8
65265	Cruise Control/Vehicle Speed	CCVS	100 ms	6	69 70 84 86 527 595 596 597 598 599 600 601 602 976 966 967 968 1237 1633	8
65266	Fuel Economy	LFE	100 ms	6	51 183 184 185	8
65270	Inlet/Exhaust Conditions 1	IC1	500 ms	6	81 102 105 106 107 112 173	8
65271	Vehicle Electrical Power	VEP	1 s	6	114 115 158 167 168	8
65272	Transmission Fluids	TF	1 s	6	123 124 126 127 177	8

**Related Functions:**

**JMP**[Symbol](#)[Inputs](#)

None

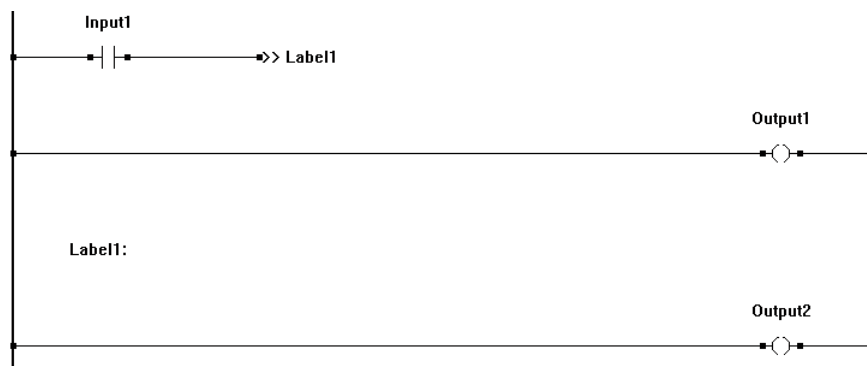
&gt;&gt; Label1

[Outputs](#)

None

[Description](#)

The JMP function allows sections of ladder to be skipped by “jumping” to another section. A LABEL must first be placed before the JMP is inserted. When the condition is true to trigger the jump, the program scan jumps to the label, skipping any ladder between the jump and label.

[Example Circuit](#)**Related Functions:** LABEL



KEYPAD

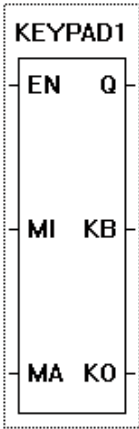
Inputs

- EN** Boolean Function Enable Input - function is disabled if EN is false.
- MI** Integer / Real - Minimum value allowed to be entered
- MA** Integer / Real - Maximum value allowed to be entered

Outputs

- Q** Boolean Function Enable Output - true for scan when ENTER is pressed.
- KB** Integer / Real Output - Buffer which holds up to 10 numeric characters, -, and a decimal point.
- KO** Integer / Real Output - When ENTER is pressed, the contents of KB are loaded into the variable connected.

Symbol



Description

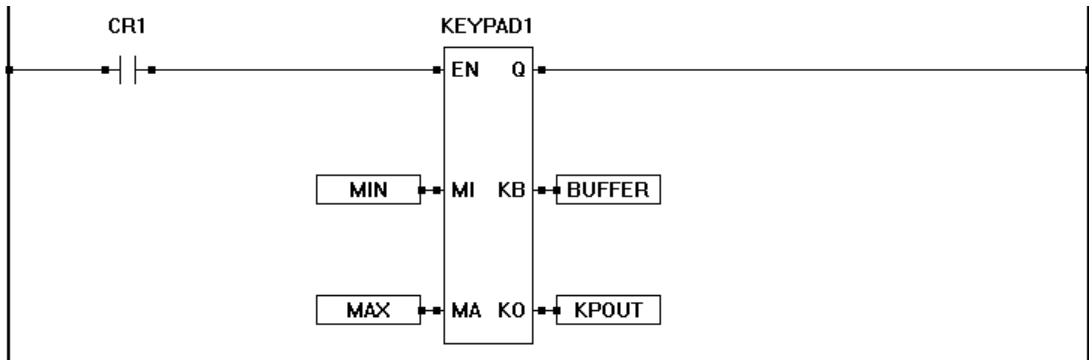
The KEYPAD function is used to allow users to input data. This function requires the Keypad feature be installed on the target's hardware and software.

The keypad may be 'monitored' in two ways. The first is using the KEYPAD function. This is useful for allowing a user to input numeric data. The second is reading individual button presses as a digital input. This is useful for menus.

Using the KEYPAD for Numeric Input

When EN is true, the function is enabled. Data is entered using numeric keypad buttons. These numeric buttons are temporarily stored in KB. When ENTER is pressed, the KB is stored in the variable connected to KO. The output Q is true for the ladder diagram scan in which the ENTER was pressed. Pressing the clear button on the keypad erases the buffer (KB).

Numeric Example



Using Discrete Inputs to Read Keys

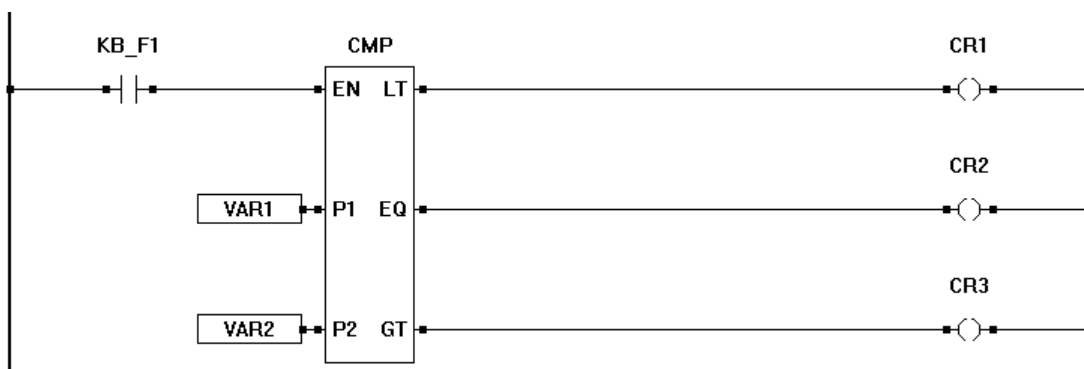
To use a digital input to read a keypress, it is necessary for at least one KEYPAD function to exist in the ladder diagram. This function 'initializes' the software to be able to use the discrete input option.

To read the keypad using a digital input (discrete), place a contact in the ladder diagram. Name the Variable as normal, select the variable type as INPUT. In the VAR I/O NUM field, enter the keypad buttons assignment. Please see the next page for button assignments. Click OK to finish placing the contact.

**Discrete Digital Input Keypad Button Assignments**

<u>I/O Assignment</u>	<u>Button Description</u>	<u>I/O Assignment</u>	<u>Button Description</u>
KB_0	Numeric 0	KB_CLEAR	Clear Button
KB_1	Numeric 1	KB_DP	Decimal Point Button
KB_2	Numeric 2	KB_+ -	+ / - Button
KB_3	Numeric 3	KB_F1	F1 Button
KB_4	Numeric 4	KB_F2	F2 Button
KB_5	Numeric 5	KB_F3	F3 Button
KB_6	Numeric 6	KB_F4	F4 Button
KB_7	Numeric 7	KB_UP	Up Button
KB_8	Numeric 8	KB_DOWN	Down Button
KB_9	Numeric 9	KB_ENTER	Enter Button

Proper care must be taken in the ladder diagram so that KEYPAD and discrete inputs do not interfere with each other's operation.

**Discrete Digital Input Example****Related Functions:**

**LABEL****Symbol****Inputs**

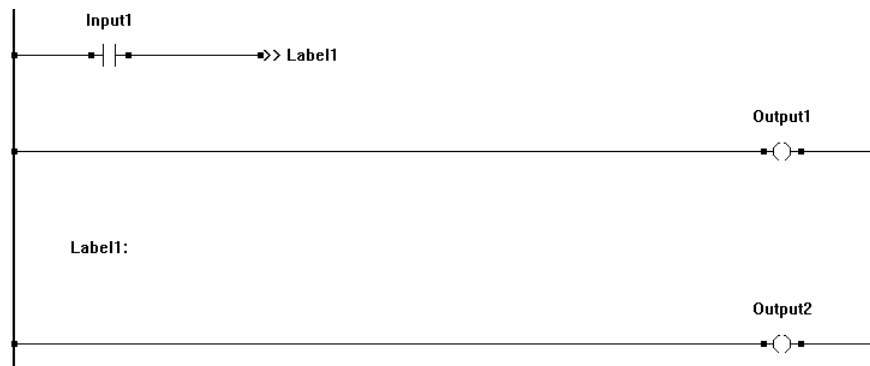
None

**Label1:****Outputs**

None

**Description**

The LABEL function works with the JMP function to skip ladder diagram sections. A LABEL must be placed first, then the JMP inserted. When the condition is true to trigger the jump, the program scan “jumps” to the LABEL, skipping any ladder between the jump and label.

**Example Circuit****Related Functions:** JMP

**LATCH (coil)**[Symbol](#)[Inputs](#)

None

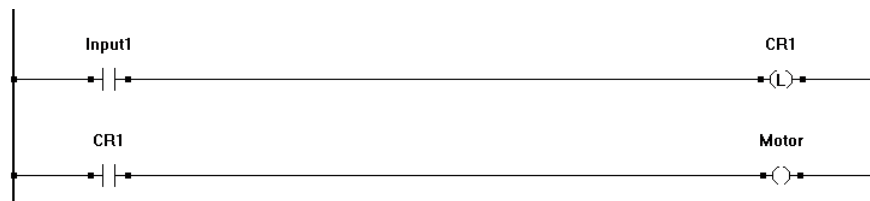
-(L)-

[Outputs](#)

None

[Description](#)

The LATCH coil acts like a direct coil, except once the coil is energized, it will maintain its energized state until the UNLATCH coil is triggered.

[Example Circuit](#)

**Related Functions:** UNLATCH, DIRECT COIL, INVERTED COIL

## LCD\_CLEAR

### Symbol

#### Inputs

**EN** Boolean Function Enable Input - sensed on rising edge.

**LCD\_CLEAR**



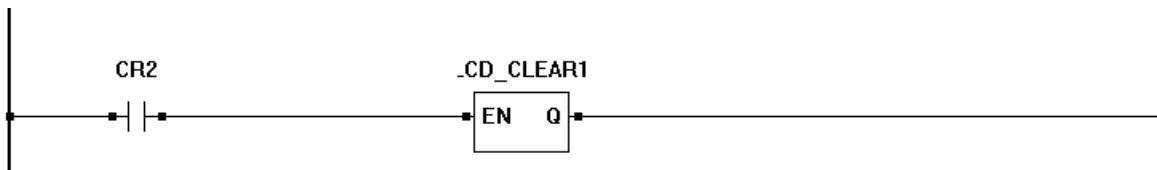
#### Outputs

**Q** Boolean Output - true the function is enabled.

#### Description

When the EN input detects a rising edge, the LCD Display is set to be cleared. The LCD display is cleared and updated at the END of the ladder scan.

#### Example Circuit



**Related Functions:** LCD\_PRINT

## LCD\_PRINT

### Symbol

#### Inputs

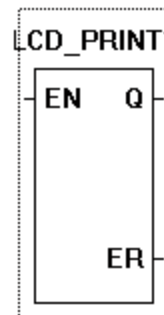
**EN** Boolean Function Enable Input - function is disabled if EN is false. EN is rising edge sensitive.

**Others as dynamically required**

#### Outputs

**Q** Boolean Function Status Output - Set true at completion of printing

**ER** Integer Error Output - Set to two if the print string is larger than the display



#### Description

The LCD\_PRINT function is the print block for printing data to the LCD Display. The LCD Display feature must be installed on the target prior to using this function block.

When the EN input senses a rising edge, the block prepares its text that was provided when the LCD\_PRINT function was placed and marks it to update at the end of the current ladder scan. The Q output is set true when the print is completed. The ER output is set true if the printed data is larger than the LCD will display.

#### Function Block

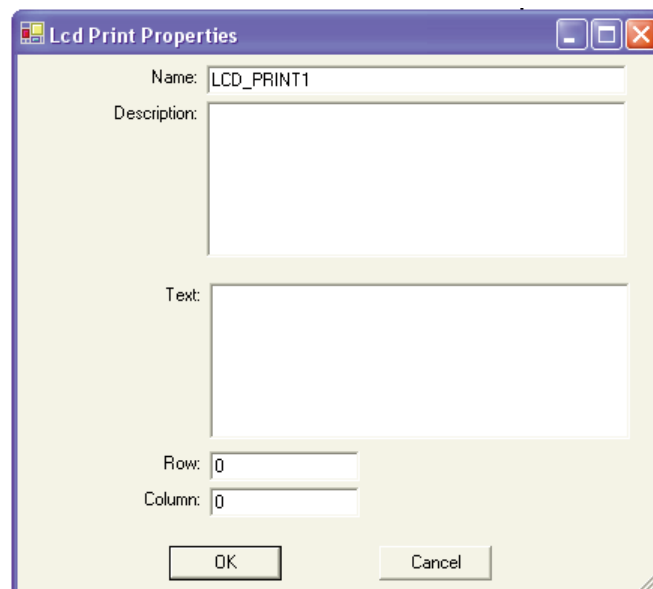
**Name:** Name of the function.

**Description:** A description of what the function does.

**Text:** The formatted text is placed here. This is what will be transmitted.

**Row:** Row to locate the start of the print.

**Column:** Column to locate the start of the print.



#### Text / Message Formatting

The LCD\_PRINT function text is formatted per ANSI C "printf". Variables as well as text may be printed. These variables must be formatted correctly. As variables are added to the 'text string' the function block will automatically add the appropriate input for the variables.

Text / Message Formatting

TEXT

Text is entered exactly as the message is intended.

VARIABLES

Variables are placed in the text using flags and print specification fields. The following is the configuration for adding variables to the text.

**%flag width .precision**                      Example Text: OIL PSI %-3d

- % - identifies the beginning of a variable or other type of text entry
- flag - This flag is optional. Use the following flags to change the way data is transmitted.

Flag	Description
-	Left align the variable within the specified 'width'. Default is align right.
0	If width is prefixed with 0, leading zeros are added until the minimum width is reached. If 0 and - are used together, the 0 is ignored. If 0 is specified in an integer format, the 0 is ignored.

- width - This flag is optional. Width is the number of characters that will be printed (total).
- .precision - This flag is optional. The precision is the number of digits after the decimal point when using REAL variables.

VARIABLE FORMATS

Variables are formatted based on the variable type. The following are supported variable types and their format.

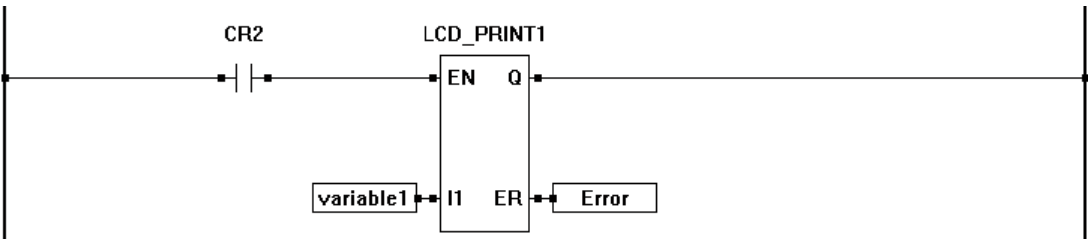
%d	Signed Integer	%X	Upper Case Hexidecimal
%u	Unsigned Integer	%f	Real or Float Variable
%x	Lower Case Hexidecimal	%b	binary
%o	Octal		

OTHER SPECIAL CHARACTERS / FORMATS

Print	Use	Print	Use
%	%%	OFF / ON	%O
Boolean 0 or 1	%d	FALSE / TRUE	%T

Examples:	<b>Format</b>	<b>Result</b>	<b>Format</b>	<b>Result</b>
	OIL: %d	OIL: 25	OIL: %04d	OIL: 0025
	LS1: %T	LS1: TRUE	LS1: %O	LS1: OFF
	TEMP: %6.2f	TEMP: 234.12	TEMP: %3.f	TEMP: 234

Example Circuit



Related Functions: LCD\_CLEAR

LESS THAN (<)

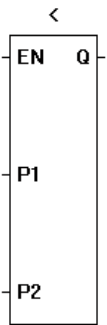
Inputs

- EN Boolean Function Enable Input - function is disabled if EN is false.
- Px Integer/Real Input - Input numbers for comparison.

Outputs

- Q Boolean Output - true when less than conditions are met.

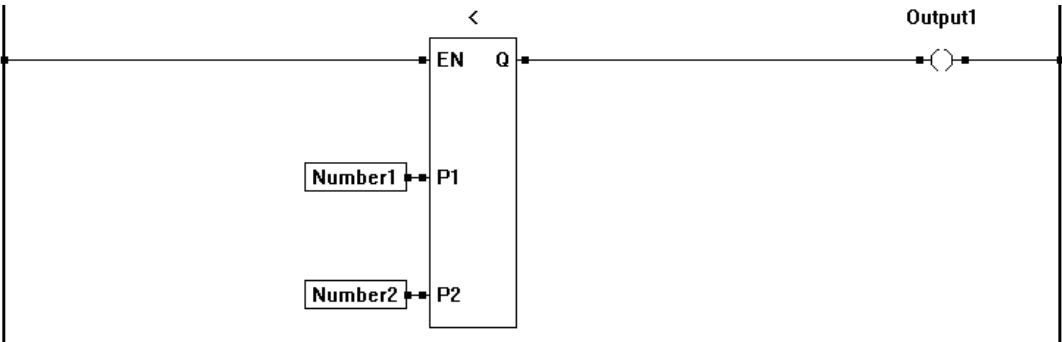
Symbol



Description

The LESS THAN provides an if less than comparison for the Px inputs. The number of inputs is specified when the object is placed. The output (Q) is true if P1 is less than P2 and P2 is less than P3 and so on. The enable (EN) must be true for the LESS THAN function to be enabled.

Example Circuit



Related Functions: =, > <=, >=, <>



LESS THAN OR EQUAL TO (<=)

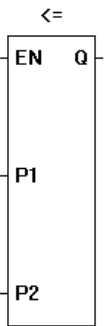
Inputs

- EN Boolean Function Enable Input - function is disabled if EN is false.
- Px Integer/Real Input - Input numbers for comparison.

Outputs

- Q Boolean Output - true when less than or equal to conditions are met.

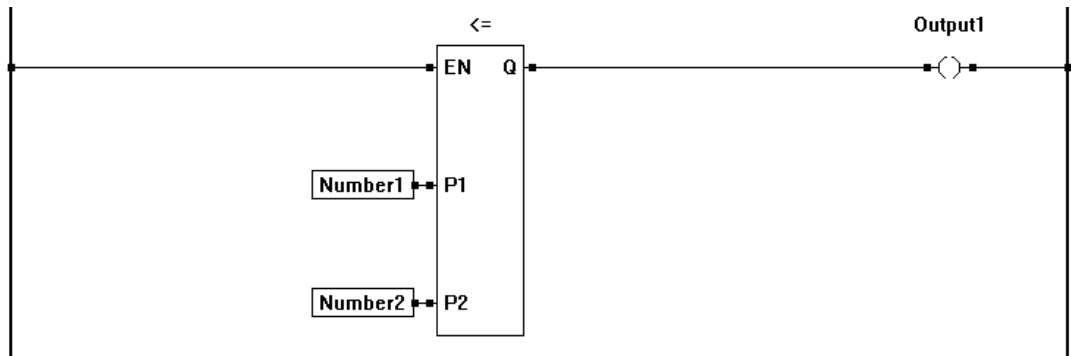
Symbol



Description

The LESS THAN OR EQUAL TO provides an if less than or equal to comparison for the Px inputs. The number of inputs is specified when the object is placed. The output (Q) is true if P1 is less than or equal to P2 and P2 is less than or equal to P3 and so on. The enable (EN) must be true for the LESS THAN OR EQUAL TO function to be enabled.

Example Circuit



Related Functions: =, >, <, >=, <>

LIMIT

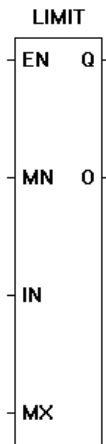
Inputs

- EN** Boolean Function Enable Input - function is disabled if EN is false.
- MN** Integer/Real Input - Minimum limit value.
- IN** Integer/Real Input - Actual input.
- MX** Integer/Real Input - Maximum limit value.

Outputs

- Q** Boolean Function Enable Output - true when function is enabled.
- O** Integer/Real Output - Actual, Minimum or Maximum value.

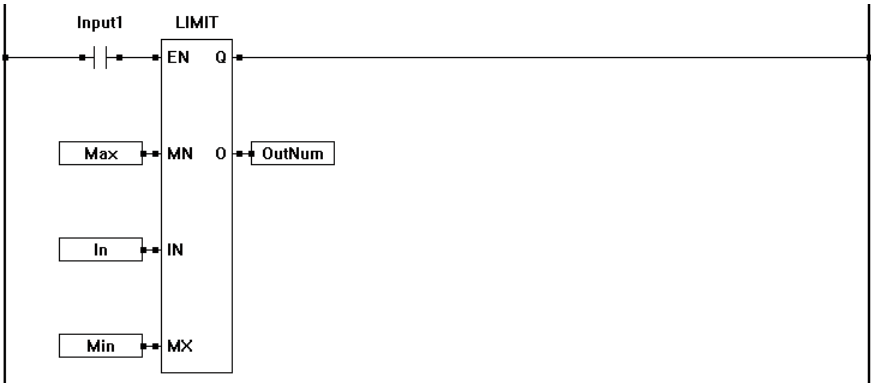
Symbol



Description

The LIMIT function provides minimum and maximum limited output for the input (IN). The function compares the input (IN). If it is greater than the maximum (MX), then the output (O) is equal to the maximum (MX). If it is less than the minimum (MN) then the output (O) is equal to the minimum (MN). If it is in between the maximum and minimum, then the output (O) is equal to the actual input (IN). The enable (EN) must be true for the LIMIT function to be enabled.

Example Circuit



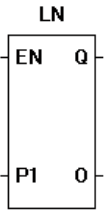
Related Functions: CMP, HYSTER, MUX

LN

Inputs

- EN Boolean Function Enable Input - function is disabled if EN is false.
- P1 Real Input - Output is natural logarithm of this number.

Symbol



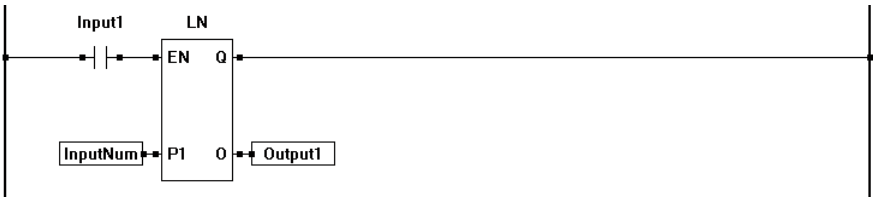
Outputs

- Q Boolean Function Enable Output - true when function is enabled.
- O Real Output - natural logarithm of the P1 input.

Description

The LN function provides the natural logarithm of the P1 input. The output (O) is the natural logarithm of the P1 input. The enable (EN) must be true for the LN function to be enabled.

Example Circuit



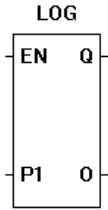
Related Functions: EXP, EXPT, MOD, SQRT, LOG

LOG

Inputs

- EN Boolean Function Enable Input - function is disabled if EN is false.
- P1 Real Input - Output is logarithm (base 10) of this input.

Symbol



Outputs

- Q Boolean Function Enable Output - true when function is enabled.
- O Real Output - logarithm (base 10) of the P1 input.

Description

The LOG function calculates the logarithm (base 10) of the P1 input. The output (O) is the cacluated base 10 (logarithm) value of the P1 input. The enable (EN) must be true for the LOG function to be enabled.

Example Circuit



Related Functions: EXP, EXPT, MOD, SQRT, LN

MAVG

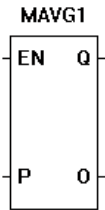
Inputs

- EN Boolean Function Enable Input - function is disabled if EN is false.
- P Integer/Real Input - Output is moving average of this input.

Outputs

- Q Boolean Function Enable Output - true when function is enabled.
- O Integer/Real Output - Moving average of P input.

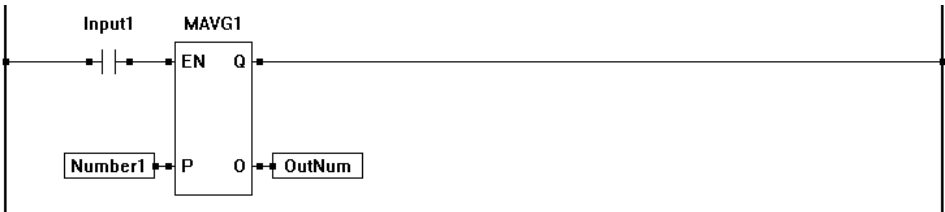
Symbol



Description

The MAVG function calculates the moving average of the P input. The number of samples are specified when the object is placed. The output (O) is the calculated moving average value of the P1 input. The enable (EN) must be true for the MAVG function to be enabled. When EN is true, the output is the moving average. When EN is false, the output is equal to the P input.

Example Circuit



Related Functions: AVG

MAX

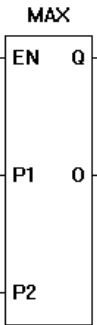
Inputs

- EN Boolean Function Enable Input - function is disabled if EN is false.
- Px Integer/Real Input - Output is equal to the largest input.

Outputs

- Q Boolean Function Enable Output - true when function is enabled.
- O Integer/Real Output - Equal to the largest Px input value.

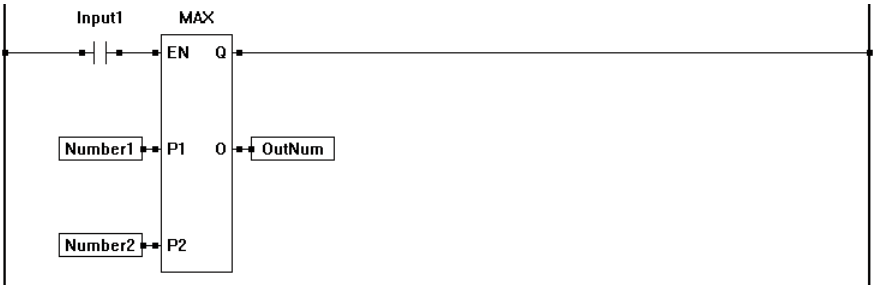
Symbol



Description

The MAX function outputs the largest of the Px input values. The number of inputs is specified when the object is placed. The enable (EN) must be true for the MAX function to be enabled.

Example Circuit



Related Functions: MIN

MIN

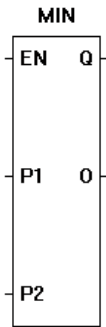
Inputs

- EN Boolean Function Enable Input - function is disabled if EN is false.
- Px Integer/Real Input - Output is equal to the smallest input.

Outputs

- Q Boolean Function Enable Output - true when function is enabled.
- O Integer/Real Output - Equal to the smallest Px input value.

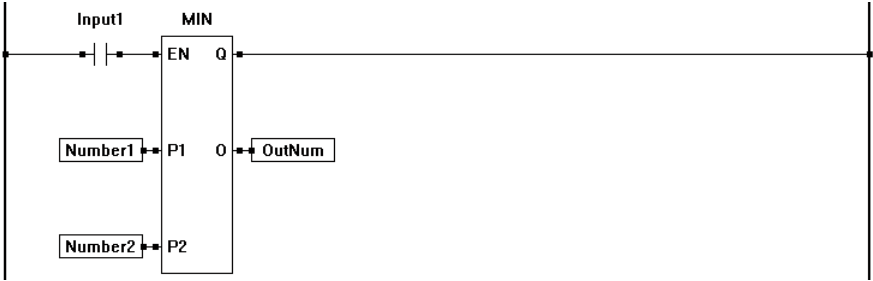
Symbol



Description

The MIN function outputs the smallest of the Px input values. The number of inputs is specified when the object is placed. The enable (EN) must be true for the MIN function to be enabled.

Example Circuit



Related Functions: MAX

MOD

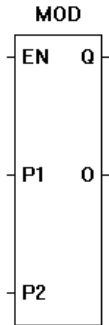
Inputs

- EN Boolean Function Enable Input - function is disabled if EN is false.
- P1 Integer Input - Dividend (# to be divided)
- P2 Integer Input - Divisor (# to divide by)

Outputs

- Q Boolean Function Enable Output - true when function is enabled.
- O Integer Output - Modulo (remainder) of P1, P2 math division.

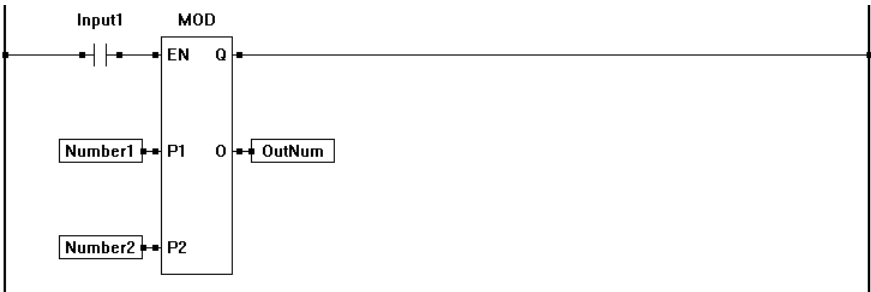
Symbol



Description

The MOD function calculates the modulo (remainder) of the division using the inputs P1 and P2. The P2 number should be greater than zero (zero or less than zero will cause the function to return a -1 for the output). The enable (EN) must be true for the MOD function to be enabled.

Example Circuit



Related Functions: DIV



MULT

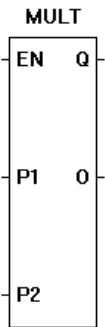
Inputs

- EN Boolean Function Enable Input - function is disabled if EN is false.
- Px Integer/Real Input - Output is equal to the multiplication of all inputs.

Outputs

- Q Boolean Function Enable Output - true when function is enabled.
- O Integer/Real Output - Equal to the multiplication of all Px inputs.

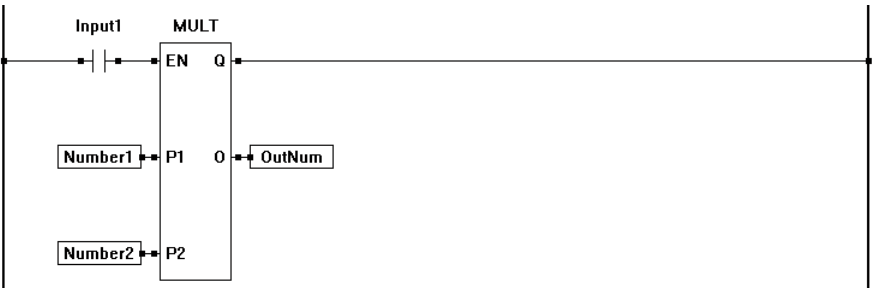
Symbol



Description

The MULT function multiplies all of the Px input together. The number of inputs is specified when the object is placed. The output (O) provides the result of the multiplication. The enable (EN) must be true for the MULT function to be enabled.

Example Circuit



Related Functions: ADD, SUB, DIV

MUX

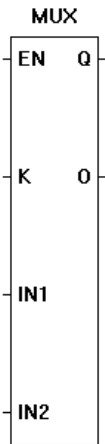
Inputs

- EN Boolean Function Enable Input - function is disabled if EN is false.
- INx Integer/Real Input - Selectable inputs.
- K Integer Input - Selection number of input to multiplex.

Outputs

- Q Boolean Function Enable Output - true when function is enabled.
- O Integer/Real Output - Value of selected input INx.

Symbol



Description

The MUX function multiplexes the INx inputs onto one output (O). The number of inputs is specified when the object is placed. The output (O) provides the value of the selected input. The selection input (K) determines the number of the input that will be present on the output. The enable (EN) must be true for the MUX function to be enabled.

Example Circuit

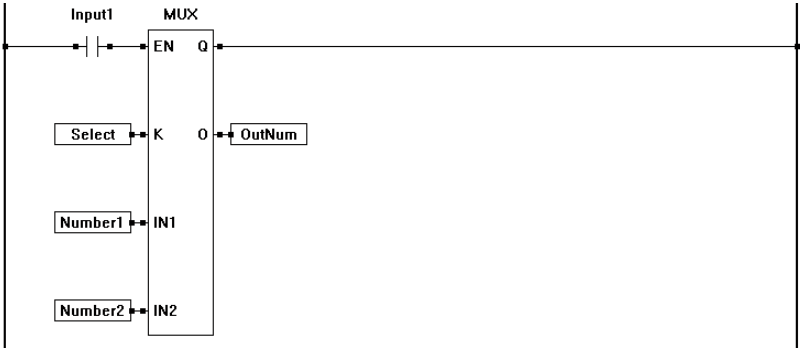


Table of Operation

Input Number	Input Value	Selector Value	Output Value
IN1	100	4	725
IN2	250	3	375
IN3	375	2	250
IN4	725	1	100

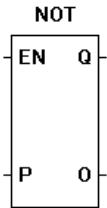
Related Functions: SEL

NOT

Inputs

- EN Boolean Function Enable Input - function is disabled if EN is false.
- P1 Integer Input - Output is one's complement of this input.

Symbol



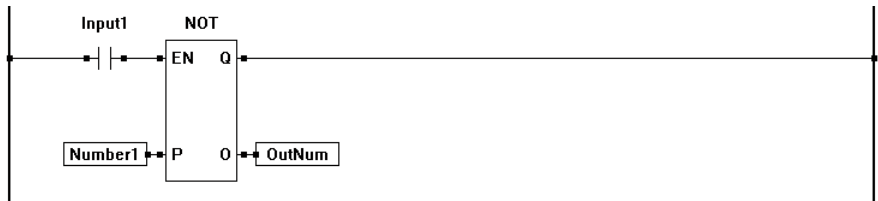
Outputs

- Q Boolean Function Enable Output - true when function is enabled.
- O Integer Output - One's complement of P1 Input.

Description

The NOT function provides a one's complement (bit to bit negation) of the P input. The output (O) provides the one's complement. The enable (EN) must be true for the NOT function to be enabled.

Example Circuit



Related Functions: OR, XOR

NOT EQUAL TO (<>)

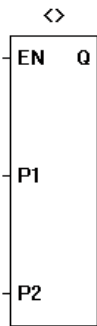
Inputs

- EN Boolean Function Enable Input - function is disabled if EN is false.
- Px Integer/Real Input - Input numbers for comparison.

Outputs

- Q Boolean Output - true when not equal to conditions are met.

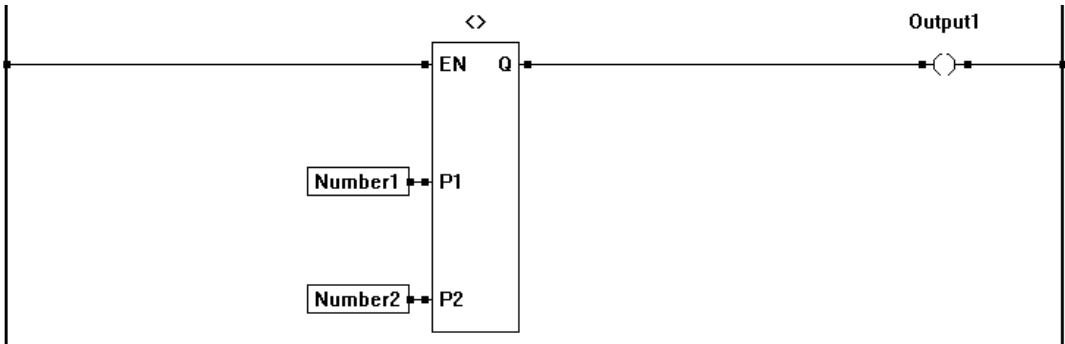
Symbol



Description

The NOT EQUAL TO provides an if greater than or less than comparison for the Px inputs. The number of inputs is specified when the object is placed. The output (Q) is true if P1 is not equal to P2 and P2 is not equal to P3 and so on. The enable (EN) must be true for the NOT EQUAL TO function to be enabled.

Example Circuit



Related Functions: <, >, <=, >=, =

**OPTICAN\_NODESTATUS**[Symbol](#)[Inputs](#)

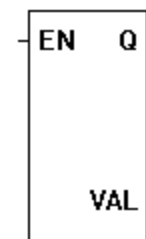
**EN** Boolean Function Enable Input - function is disabled if EN is false.

[Outputs](#)

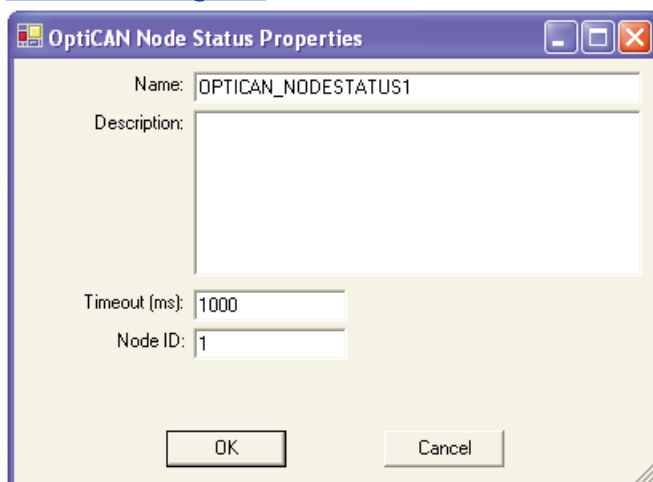
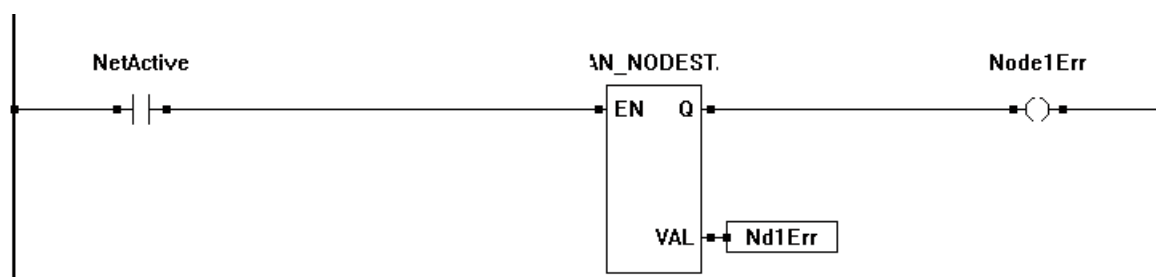
**Q** Boolean Output - true when node status messages are being received, false when no node status messages are received within the specified timeout.

**VAL** Integer Output - 32-bit number. Lower 16-bits represent the node status and the upper 16-bits represent the actual error code.

OPTICAN\_NODESTATUS

[Description](#)

The OPTICAN\_NODESTATUS function 'listens' for OK of the node status register for the specified address over the OptiCAN network. When placing the function, the NODE ID is specified as well as the Timeout. An optional description may be included. The function block will 'listen' for the node status register broadcast of the Node ID and update VAL and Q accordingly. The Timeout value is the duration that the function block will 'listen' without receiving a status without generating an Error. See EZ LADDER Manual, Section 16 - OptiCAN Network for more information regarding using the function block and general OptiCAN networking.

[Function Dialog Box](#)[Example Circuit](#)

**Related Functions:** OPTICAN\_TXNETMSG

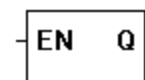
**OPTICAN\_TXNETMSG**[Symbol](#)[Inputs](#)

**EN** Boolean Function Enable Input - function is disabled if EN is false.

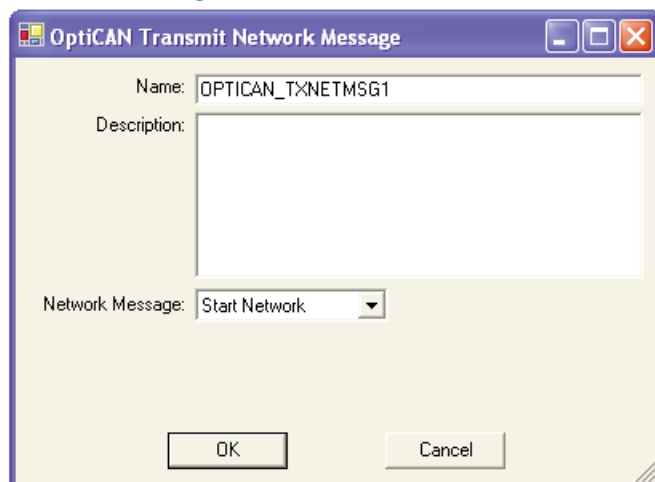
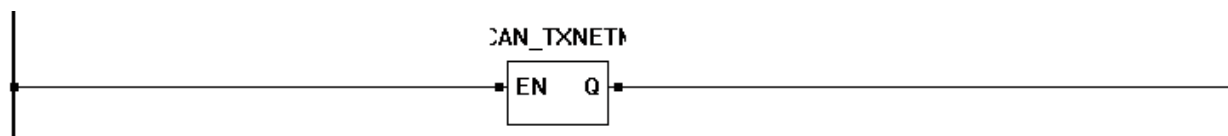
[Outputs](#)

**Q** Boolean Output - true when function is enabled.

OPTICAN\_TXNETMSG

[Description](#)

The OPTICAN\_TXNETMSG function broadcasts the “Start Network, Stop Network and Reset Network” commands over the OptiCAN network. This function block globally broadcasts, therefore affecting all connected nodes. A ‘Start Network’ command must be broadcast after power up to start the OptiCAN network nodes communications. When placing the function, a dialog box provides the selection of the type of command to send and an optional description box. See EZ LADDER Manual, Section 16 - OptiCAN Network for more information regarding using the function block and general OptiCAN networking.

[Function Dialog Box](#)[Function Example](#)

**Related Functions:** OPTICAN\_NODESTATUS

OR

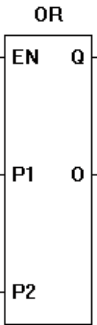
Inputs

- EN Boolean Function Enable Input - function is disabled if EN is false.
- P1 Integer Input - Output is bitwise OR of inputs.
- P2 Integer Input - Output is bitwise OR of inputs.

Outputs

- Q Boolean Function Enable Output - true when function is enabled.
- O Integer Output - Bitwise OR of P1, P2 Inputs.

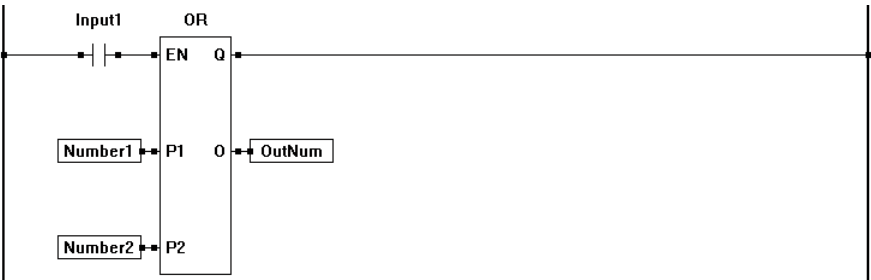
Symbol



Description

The OR function provides a bitwise OR function of the P1 and P2 inputs. The enable (EN) must be true for the OR function to be enabled. The Q output is true when the OR function is enabled.

Example Circuit



Related Functions: AND, XOR, NOT

PID

Inputs

- EN** Boolean Function Enable Input - function is disabled if EN is false.
- SP** Real Input - Setpoint, the actual control value that is required to be met.
- PV** Real Input - Process Variable, actual input to be controlled.
- KP** Real Input - Proportional Gain Input
- KI** Real Input - Integral Gain Input
- KD** Real Input - Derivative Gain Input
- IO** Real Input - Initial Output (Value that PID output is initialized to).

Outputs

- Q** Boolean Function Enable Output - true when function is enabled.
- CO** Real Output - Control Output, the calculated output of the PID function.
- ER** Real Output - Error, the calculated error of the PID function.

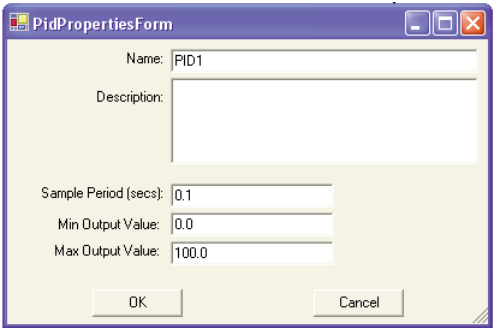
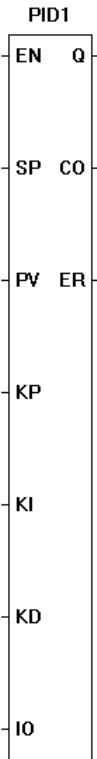
Description

The PID function provides an easy to use PID control algorithm. Specific PID information is required when the function is placed (see below) as well as the PID inputs (see above). The Q is true when the function is enabled. The CO (Control Output) is the output calculated by the PID (new control signal). The ER (Error) is the error calculation of the PID (SP-PV). The PID function is defined by the difference Equation:

$$u(n) = u(n-1) + K_p[e(n) - e(n-1)] + K_i [T * e(n)] + (K_d / T)[e(n) - 2 * e(n-1) + e(n-2)]$$

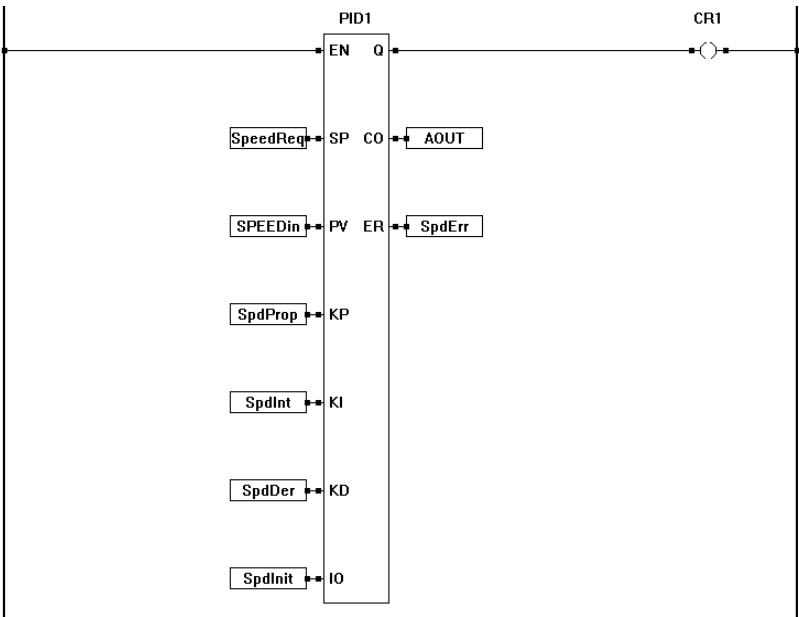
Where:  $u(n)$  = PID Output       $K_p$  = Proportional Gain       $K_i$  = Integral Gain  
 $K_d$  = Derivative Gain       $e(n)$  = Error (Setpoint - Process Variable)       $T$  = Sample Period

Symbol



- Name:** Name of the PID function
- Description:** Enter a description of the PID use.
- Sample Period (Secs):** The sample period in seconds (Min = .01S, Max = 86,400S), sample period resolution = 50µS.
- Minimum Output Value:** Minimum PID Output value allowed.
- Maximum Output Value:** Maximum PID Output value allowed.

Example Circuit



Related Functions:



PWM

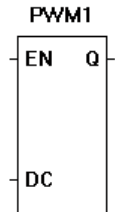
Inputs

- EN Boolean Function Enable Input - function is disabled if EN is false.
- DC Integer/Real Input - Duty Cycle of the PWM Channel Output, 0-100 representing 0% to 100%

Outputs

- Q Boolean Function Enable Output - true when function is enabled.

Symbol

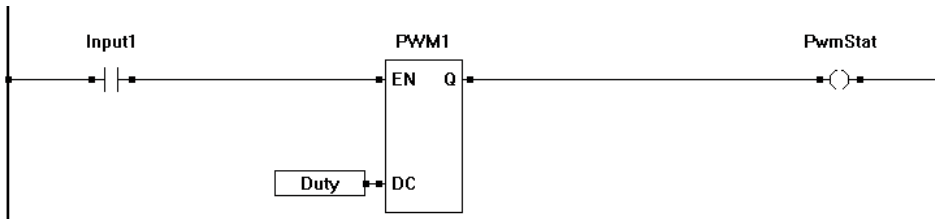


Description

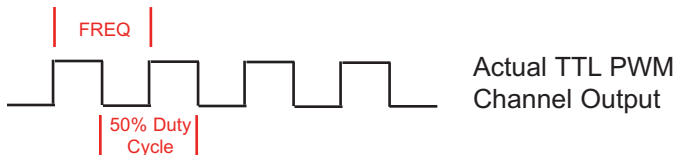
The PWM function controls the function of a **hardware** PWM output channel (this channel is specified when the function is placed). When the EN is true, the **hardware** PWM channel outputs a square wave with the specified duty cycle (DC function input) at the frequency preprogrammed (this frequency is determined by the PWM channel and is configured when the PWM channel is installed in the target settings menu unless the PWM\_FREQ function overrides this frequency with it's own). The Q output is true when the function is enabled.

When the PWM function is placed, you must specify the actual **hardware** PWM channel that the function will control and the Polarity (Starting Low will cause the PWM channel to start with a TTL low, Starting High will cause the PWM channel to start with a TTL high).

Example Circuit



Frequency & Duty Cycle



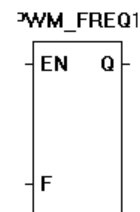
Related Functions: PWM\_FREQ

## PWM\_FREQ

### Symbol

#### Inputs

- EN** Boolean Function Enable Input - function is disabled if EN is false.  
**F** Integer Input - New Frequency to apply to PWM Output channel.



#### Outputs

- Q** Boolean Function Enable Output - true when the F Input frequency is actually being used on the PWM Output channel.

#### Description

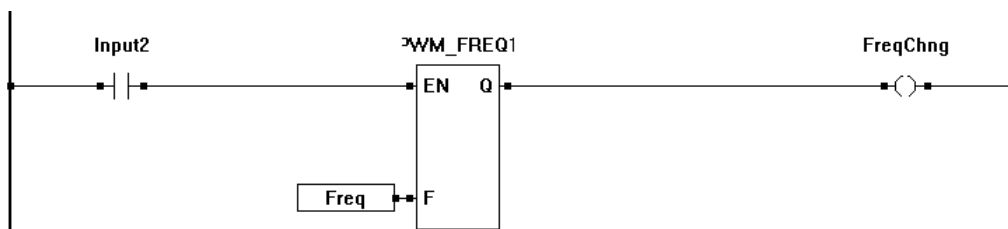
The PWM\_FREQ function controls the frequency of a **hardware** PWM output channel (this channel is specified when the function is placed). When the EN sees a low to high transition, the **hardware** PWM channel's frequency is changed from it's current value (either from when the PWM channel was installed using the Target..Settings menu or a PWM\_FREQ function).

The PWM\_FREQ only changes the **hardware** PWM channel's frequency with a low to high transition on EN. This frequency will be maintained regardless of the EN state. The only time this frequency will change again is when the actual frequency input variable (input F) changes and the EN detects another low to high transition. Q is true during the ladder diagram scan when the frequency is newly applied. All other times, the Q output is low.

When the PWM function is placed, you must specify the actual PWM channel group (CLK A or CLK B) that the function will change the frequency to.

Note: If an invalid frequency is input to F, then the Q Output will remain low as well as the actual PWM output.

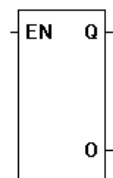
#### Example Circuit



**Related Functions:** PWM

**RANDOM****Symbol****Inputs**

**EN** Boolean Function Enable Input - function is disabled if EN is false.

**RANDOM****Outputs**

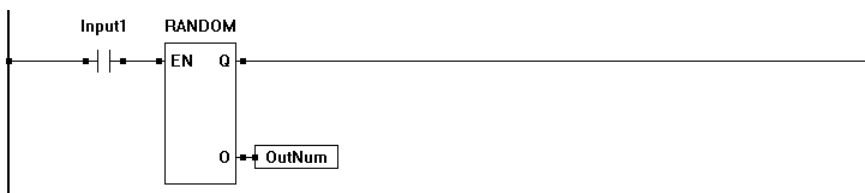
**Q** Boolean Function Enable Output - true when function is enabled.

**O** Integer Output - Random Number.

**Description**

The RANDOM functions provides a random number based on the SEED (function) value. The enable (EN) must be true for the RANDOM function to be enabled. The Q output is true when the RANDOM function is enabled. The output (O) is the random number.

The RANDOM function is designed to be used with the SEED function. Without the SEED, the output will not be random.

**Example Circuit**

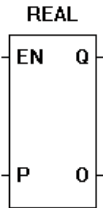
**Related Functions:** SEED

REAL

Inputs

- EN Boolean Function Enable Input - function is disabled if EN is false.
- P Boolean/Integer/Timer Input - Output is real of this input.

Symbol



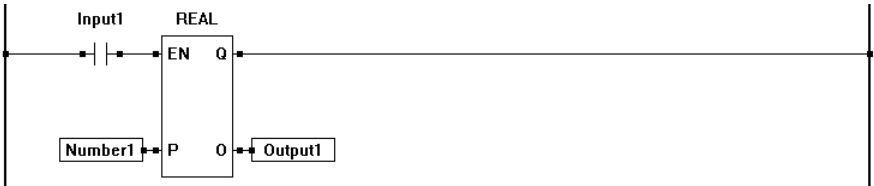
Outputs

- Q Boolean Function Enable Output - true when function is enabled.
- O Real Output - Conversion of the P input. If the input type is a timer, then the output is a real representation in milliseconds of the input value.

Description

The REAL function converts the input (P) into an real ouput (O). The enable (EN) must be true for the REAL function to be enabled. The Q output is true when the REAL function is enabled.

Example Circuit



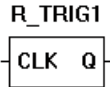
Related Functions: BOOL, INTEGER

R\_TRIG

Inputs

CLK Boolean Function Enable Input - detects rising edge of CLK.

Symbol



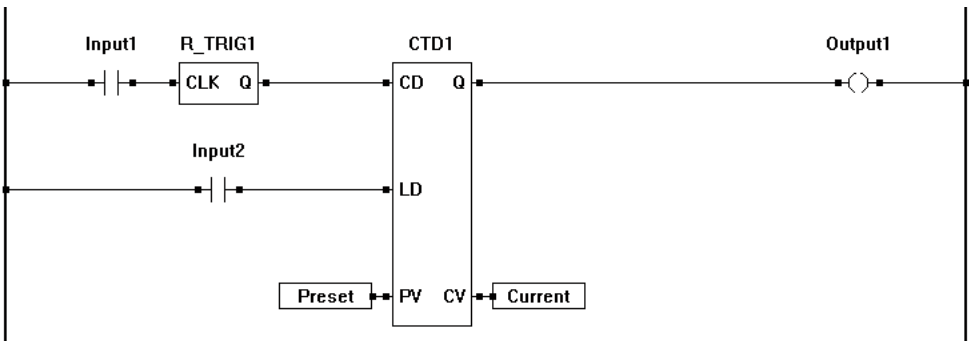
Outputs

Q Boolean Output - Pulsed Output, true for one scan when CLK detects a rising edge.

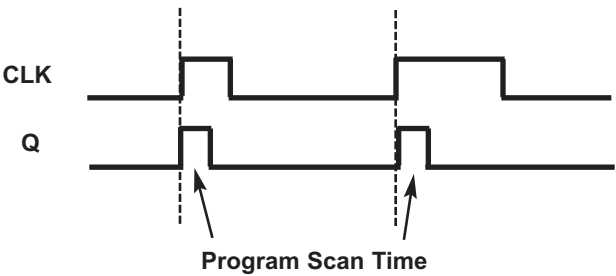
Description

The R\_TRIG is a function that may be used to trigger another function on the rising edge of a transition. When the CLK detects a false to true transition, the output (Q) is energized (for one scan of the program only).

Example Circuit



Timing Diagram



Related Functions: F\_TRIG

ROL

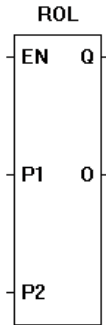
Inputs

- EN Boolean Function Enable Input - function is disabled if EN is false.
- P1 Integer Input - Base number that will have bit rotations
- P2 Integer Input - Number of left one-bit rotations to occur for P1

Outputs

- Q Boolean Function Enable Output - true when function is enabled.
- O Integer Output - Output result of bit rotation.

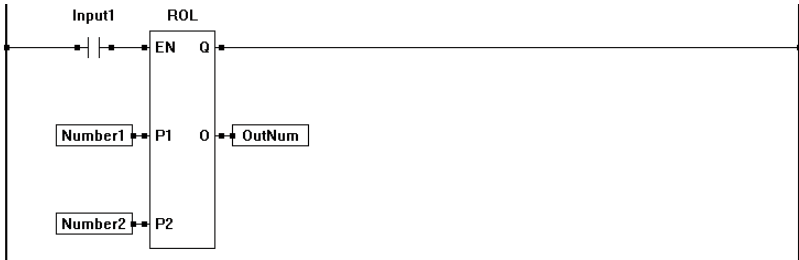
Symbol



Description

The ROL function provides a left-bit rotation of the P1 input. P2 specifies the number of one-bit rotations. The P1 number is a integer representation of a binary number. The P2 number is an integer representation of the number of binary rotations (shifts) to occur to P1. The actual bit only rotates when the maximum number is reached (example: 32 bit rotation to the input number 1).The enable (EN) must be true for the ROL function to be enabled. The Q output is true when the ROL function is enabled. The O Output is the rotated number (represented in integer form).

Example Circuit



Related Functions: ROR

ROR

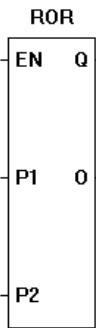
Inputs

- EN Boolean Function Enable Input - function is disabled if EN is false.
- P1 Integer Input - Base number that will have bit rotations
- P2 Integer Input - Number of right one-bit rotations to occur for P1

Outputs

- Q Boolean Function Enable Output - true when function is enabled.
- O Integer Output - Output result of bit rotation.

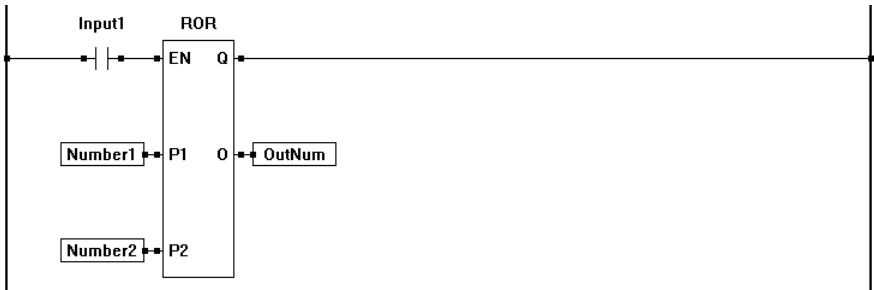
Symbol



Description

The ROR function provides a right-bit rotation of the P1 input. P2 specifies the number of one-bit rotations. The P1 number is a integer representation of a binary number. The P2 number is an integer representation of the number of binary rotations (shifts) to occur to P1. The actual bit only rotates when the minimum number is reached (example: 32 bit rotation to the input number 32). The enable (EN) must be true for the ROR function to be enabled. The Q output is true when the ROR function is enabled. The O Output is the rotated number (represented in integer form).

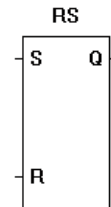
Example Circuit



Related Functions: ROL

**RS****Symbol****Inputs**

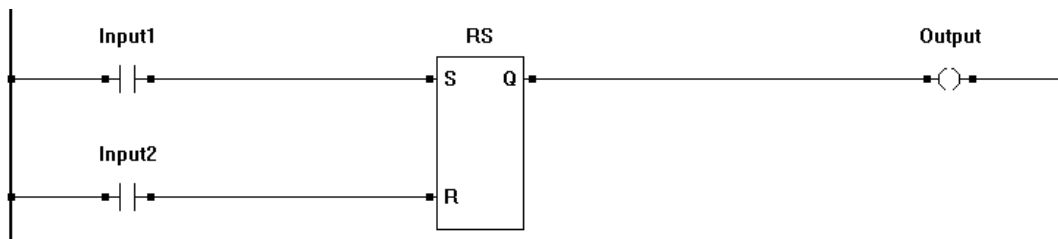
- S** Boolean Input - function SET input.  
**R** Boolean Input - function RESET input.

**Outputs**

- Q** Boolean Output - Latched output

**Description**

The RS function acts as a reset dominant bistable. If the set input (S) is true, the output (Q) is true. A true on the reset (R) input sets the output (Q) to false (regardless of the set (S) input state).

**Example Circuit****Truth Table**

SET	RESET	Q	Q RESULT
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	0

**Related Functions:** SR

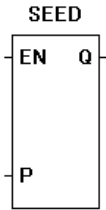


SEED

Inputs

- EN Boolean Function Enable Input - function is disabled if EN is false.
- P Integer Input - Number to base SEED generation on.

Symbol



Outputs

- Q Boolean Function Enable Output - true when function is enabled.

Description

The SEED function provides the number which the RANDOM function uses as the basis for generating a random number. The enable (EN) must be true for the SEED function to be enabled. The Q output is true when the SEED function is enabled.

Example Circuit



Related Functions: RANDOM

SEL

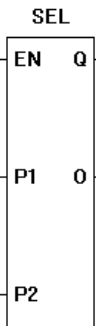
Inputs

- EN Boolean Function Enable Input - function is disabled if EN is false.
- P1 Integer/Real Input - Selectable input # 1
- P2 Integer/Real Input - Selectable input # 2

Outputs

- Q Boolean Function Enable Output - true when function is enabled.
- O Integer/Real Output - Output = selected input.

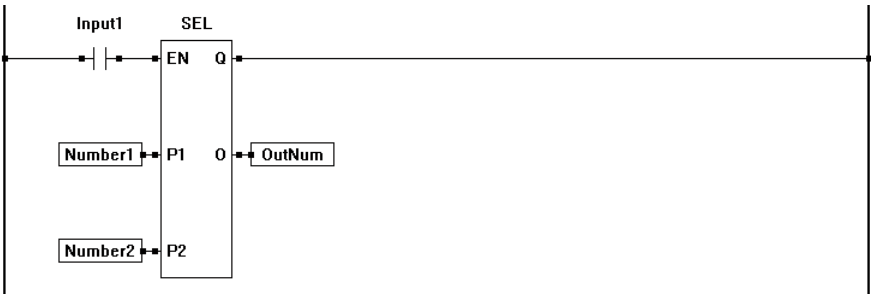
Symbol



Description

The SEL function provides selection of the P1 and P2 inputs. If enable (EN) is false, the output (O) will be equal to the input P1. If the enable (EN) is true, the output (O) will be equal to the input P2. The Q output is true when the SEL function is enabled.

Example Circuit



Related Functions: MUX

**SERIAL\_PRINT**[Symbol](#)[Inputs](#)

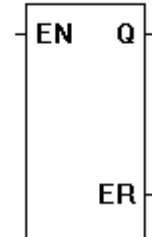
**EN** Boolean Function Enable Input - function is disabled if EN is false.  
EN is rising edge sensitive.

**Others as dynamically required**

[Outputs](#)

**Q** Boolean Function Status Output - Set true at completion of transmission

**ER** Integer Error Output - Set to one if transmit buffer is not empty when transmission starts or two if the transmit string is larger than the buffer.

**SERIAL\_PRINT**[Description](#)

The SERIAL\_PRINT function is the transmit block for sending serial information using a multi-purpose serial port. The Serial Print feature must be installed on the target prior to using this function block.

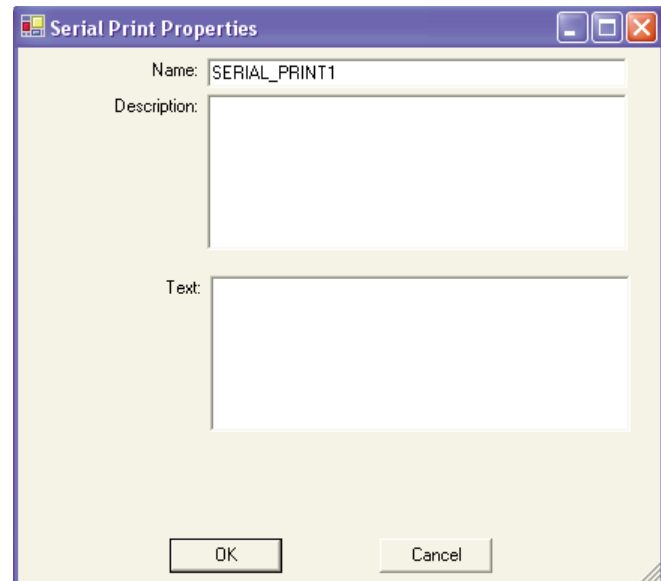
When the EN input senses a rising edge, the block begins the serial transmission of its text that was provided when the SERIAL\_PRINT function was placed. The Q output is set true when the transmission is completed. The ER output is set true if there is still data in the buffer when the function block is enabled to transmit again.

[Function Block](#)

**Name:** Name of the function.

**Description:** A description of what the function does.

**Text:** The formatted text is placed here. This is what will be transmitted.

[Text / Message Formatting](#)

The SERIAL\_PRINT function text is formatted per ANSI C "printf". The function block examples shown are for VT100 terminals. Variables as well as text may be printed. These variables must be formatted correctly. As variables are added to the 'text string' the function block will automatically add the appropriate input for the variables.

[Text / Message Formatting](#)**TEXT**

Text is entered exactly as the message is intended.

VARIABLES

Variables are placed in the text using flags and print specification fields. The following is the configuration for adding variables to the text.

**%flag width .precision**                      Example Text: OIL PSI %-3d

- % - identifies the beginning of a variable or other type of text entry
- flag - This flag is optional. Use the following flags to change the way data is transmitted.

Flag	Description
-	Left align the variable within the specified 'width'. Default is align right.
0	If width is prefixed with 0, leading zeros are added until the minimum width is reached. If 0 and - are used together, the 0 is ignored. If 0 is specified in an integer format, the 0 is ignored.

- width - This flag is optional. Width is the number of characters that will be printed (total).
- .precision - This flag is optional. The precision is the number of digits after the decimal point when using REAL variables.

VARIABLE FORMATS

Variables are formatted based on the variable type. The following are supported variable types and their format.

%d	Signed Integer	%X	Upper Case Hexidecimal
%u	Unsigned Integer	%f	Real or Float Variable
%x	Lower Case Hexidecimal	\Xxx	xx are hex characters. Prints xx character.

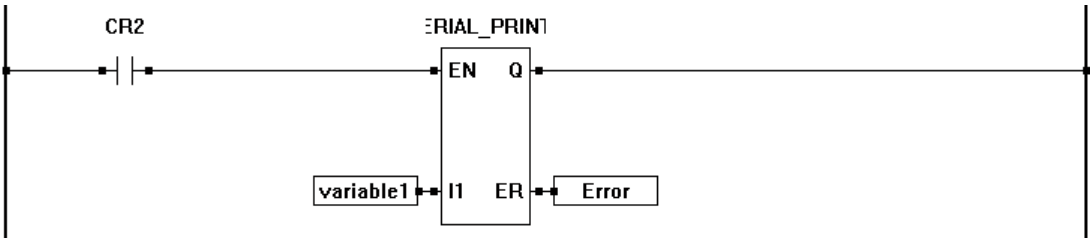
OTHER SPECIAL CHARACTERS / FORMATS

Print	Use	Print	Use
%	%%	OFF / ON	%O
Boolean 0 or 1	%d	FALSE / TRUE	%T

- To clear a VT100 screen, use: "\X1B[H\x1b[2J"
- To locate a fixed print on a VT100 screen, use: "\X1B[5,1HText". This will locate and print 'Text' at row 5, column 1.
- To locate an adjustable print on a VT100 screen (using variables): "\X1B[%d;%dHText". This will locate the row then the column using the variables defined (integers) in the order they were defined. The first integer input on the block becomes the row and the next becomes the column. At that location point, the 'Text' is printed.

Examples:	<b>Format</b>	<b>Result</b>	<b>Format</b>	<b>Result</b>
	OIL: %d	OIL: 25	OIL: %04d	OIL: 0025
	LS1: %T	LS1: TRUE	LS1: %O	LS1: OFF
	TEMP: %6.2f	TEMP: 234.12	TEMP: %3.f	TEMP: 234

Example Circuit



Related Functions:

SETDATE

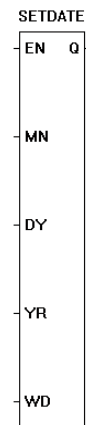
Inputs

- EN** Boolean Function Enable Input - function is disabled if EN is false.
- MN** Integer Input - Month value (1-12).
- DY** Integer Input - Day value (1-31).
- YR** Integer Input - Year value (0-99).
- WD** Integer Input - Day of the Week value (1-7).

Outputs

- Q** Boolean Function Enable Output - true when function is enabled.

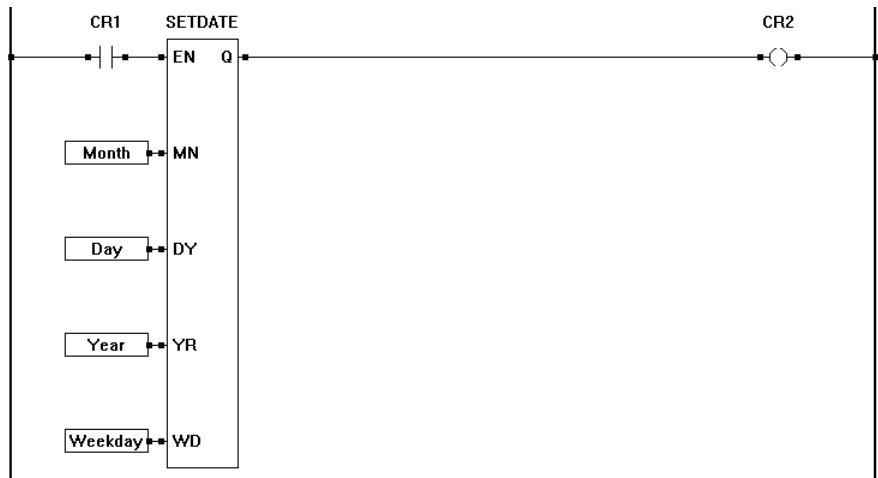
Symbol



Description

The SETDATE function sets the current date on the hardware real time clock. The date is set by using variables to apply values to each of the inputs. The enable (EN) must be true for the SETDATE function to be enabled.

Example Circuit



**Related Functions:** SETTIME, GETTIME, GETDATE

SETTIME

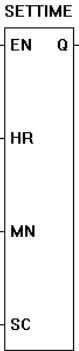
Inputs

- EN Boolean Function Enable Input - function is disabled if EN is false.
- HR Integer Input - Hours value (0-23).
- MN Integer Input - Minutes value (0-59).
- SC Integer Input - Seconds value (0-59).

Outputs

- Q Boolean Function Enable Output - true when function is enabled.

Symbol



Description

The SETTIME function sets the current time on the hardware real time clock. The time is set by using variables to apply values to each of the inputs. The enable (EN) must be true for the SETTIME function to be enabled.

Example Circuit



Related Functions: SETDATE, GETTIME, GETDATE

SHL

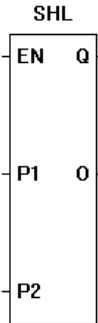
Inputs

- EN Boolean Function Enable Input - function is disabled if EN is false.
- P1 Integer Input - Number to be shifted left.
- P2 Integer Input - Number of one-bit left shifts.

Outputs

- Q Boolean Function Enable Output - true when function is enabled.
- O Integer Output - Result of the shifted input.

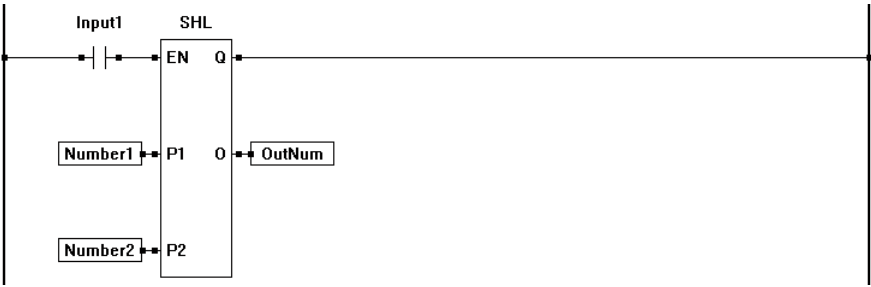
Symbol



Description

The SHL function provides a left bit shift of the P1 input. The P2 input specifies the number of one-bit left shifts. If the enable (EN) is false, the function is disabled. If the enable (EN) is true, the output (O) will be equal result of the left shifted input in integer form (1..2..4..8..16..32). A shift left when the output is 32 will cause the output to be zero (bit is shifted off). Zeros are always shifted on to the right side when a left shift occurs.

Example Circuit



Related Functions: SHR

SHR

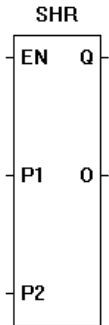
Inputs

- EN Boolean Function Enable Input - function is disabled if EN is false.
- P1 Integer Input - Number to be shifted right.
- P2 Integer Input - Number of one-bit left right.

Outputs

- Q Boolean Function Enable Output - true when function is enabled.
- O Integer Output - Result of the shifted input.

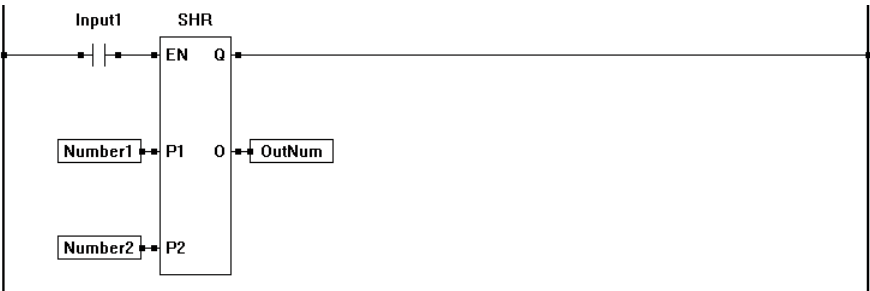
Symbol



Description

The SHR function provides a right bit shift of the P1 input. The P2 input specifies the number of one-bit right shifts. If the enable (EN) is false, the function is disabled. If the enable (EN) is true, the output (O) will be equal result of the right shifted input in integer form (32..16..8..4..2..1). A shift right when the output is 1 will cause the output to be zero (bit is shifted off). Zeros are always shifted on to the left side when a right shift occurs.

Example Circuit



Related Functions: SHL

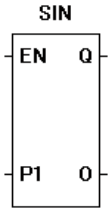


SIN

Inputs

- EN Boolean Function Enable Input - function is disabled if EN is false.
- P1 Real Input - Output is sine of this input.

Symbol



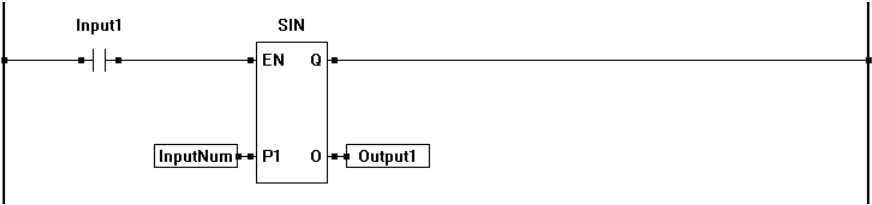
Outputs

- Q Boolean Function Enable Output - true when function is enabled.
- O Real Output - Sine value of P1 input.

Description

The SIN function provides the sine (O) from the input value (P1). The enable (EN) must be true for the SIN function to be enabled. The Q output is true when the SIN function is enabled.

Example Circuit



Related Functions: ACOS, TAN, ATAN, COS, ASIN

SI\_DISP

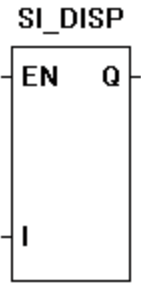
Inputs

- EN Boolean Function Enable Input - function is disabled if EN is false.
- I Integer Input - This variable value will be displayed on the Solves-It!'s display..

Outputs

- Q Boolean Function Enable Output - true when function is enabled.

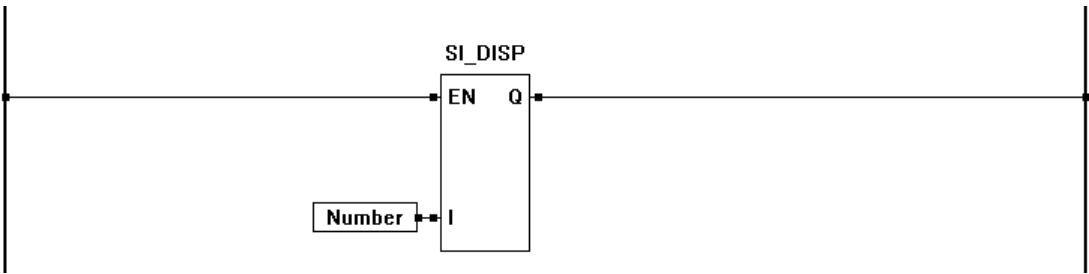
Symbol



Description

The SI\_DISP function writes integer values to the Solves-It!'s 4-digit display. When enabled, the integer value of the variable connected to the I input will be displayed.

Example Circuit

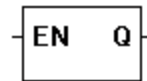


Related Functions: SI\_CLRDISP

## SI\_CLRDISP

[Symbol](#)[Inputs](#)

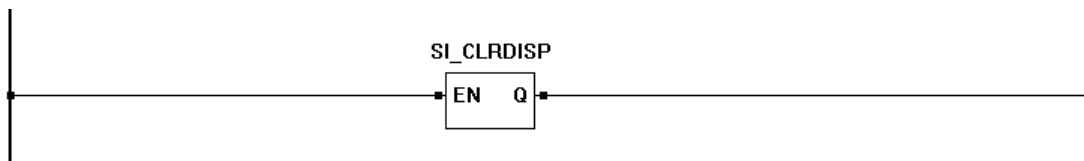
**EN** Boolean Function Enable Input - function is disabled if EN is false.

[SI\\_CLRDISP](#)[Outputs](#)

**Q** Boolean Function Enable Output - true when function is enabled.

[Description](#)

The SI\_CLRDISP function erases what is currently displayed on the Solves-It!'s 4 digit display.

[Example Circuit](#)

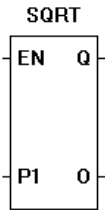
**Related Functions:** SI\_DISP

SQRT

Inputs

- EN Boolean Function Enable Input - function is disabled if EN is false.
- P1 Real Input - Output is square root of this input.

Symbol



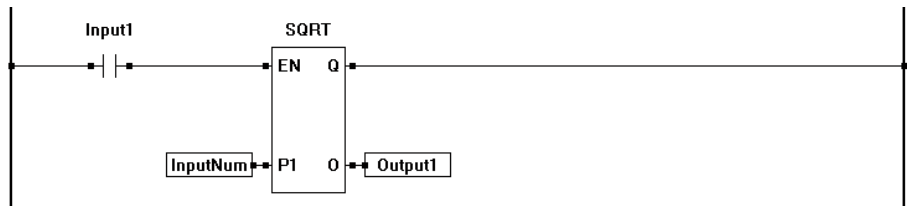
Outputs

- Q Boolean Function Enable Output - true when function is enabled.
- O Real Output - Square root value of P1 input.

Description

The SQRT function provides the square root (O) from the input value (P1). The enable (EN) must be true for the SQRT function to be enabled. The Q output is true when the SQRT function is enabled.

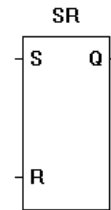
Example Circuit



**Related Functions:** MULT, DIV, MOD, EXP, LN, EXPT, LOG

**SR****Symbol****Inputs**

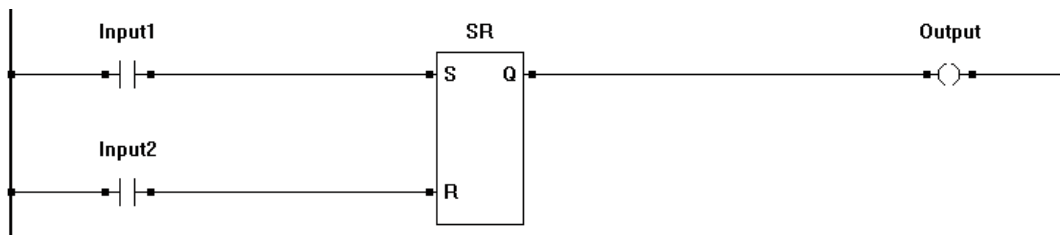
- S** Boolean Input - function SET input.  
**R** Boolean Input - function RESET input.

**Outputs**

- Q** Boolean Output - Latched output

**Description**

The SR function acts as a set dominant bistable. If the set input (S) is true, the output (Q) is true. A true on the reset (R) input sets the output (Q) to false only if the set (S) input is also false.

**Example Circuit****Truth Table**

SET	RESET	Q	Q RESULT
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

**Related Functions:** RS

SUB

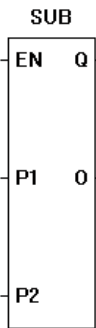
Inputs

- EN Boolean Function Enable Input - function is disabled if EN is false.
- P1 Integer/Real Input - Number to subtract P2 from.
- P2 Integer/Real Input - Number to subtract from P1.

Outputs

- Q Boolean Function Enable Output - true when function is enabled.
- O Integer/Real Output - Result of subtraction (P1-P2).

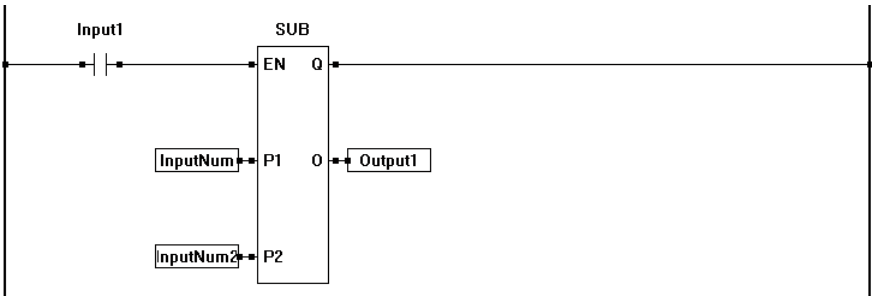
Symbol



Description

The SUB functions subtracts the P2 input from the P1 input. The output (O) is the result of the subtraction. The enable (EN) must be true for the SUB function to be enabled. The Q output is true when the SUB function is enabled.

Example Circuit



Related Functions: ADD, MULT, DIV, ABS

TAN

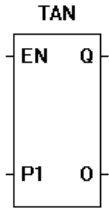
Inputs

- EN Boolean Function Enable Input - function is disabled if EN is false.
- P1 Real Input - Output is tangent of this input.

Outputs

- Q Boolean Function Enable Output - true when function is enabled.
- O Real Output - Tangent value of P1 input.

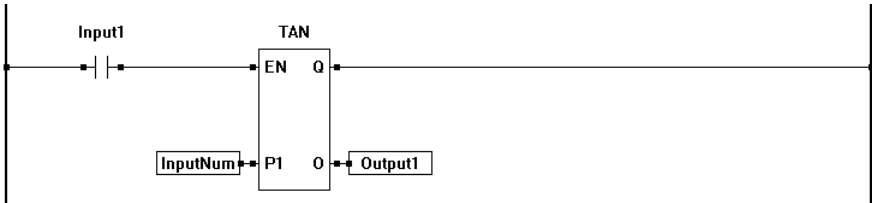
Symbol



Description

The TAN function provides the Tangent (O) from the input value (P1). The enable (EN) must be true for the TAN function to be enabled. The Q output is true when the TAN function is enabled.

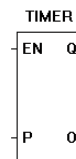
Example Circuit



Related Functions: ATAN, SIN, ASIN, COS, ACOS

**TIMER****Symbol****Inputs**

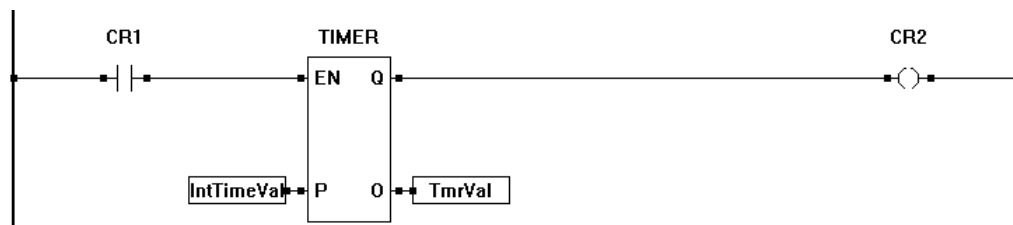
- EN** Boolean Function Enable Input - function is disabled if EN is false.  
**P** Real / Integer Input - Output Timer Value (converted based on milliseconds.)

**Outputs**

- Q** Boolean Function Enable Output - true when function is enabled.  
**O** Timer Output - Converted value of P.

**Description**

The TIMER function converts a real or integer value into a timer value. The EN enables the function and the P input is the real or integer input value to convert. The O is the timer variable type output.

**Example Circuit****Related Functions:**

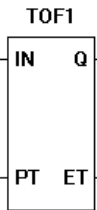


TOF

Inputs

- IN Boolean Input - Timer sense/trigger input.
- PT Timer Input - Maximum time value (preset timer value).

Symbol



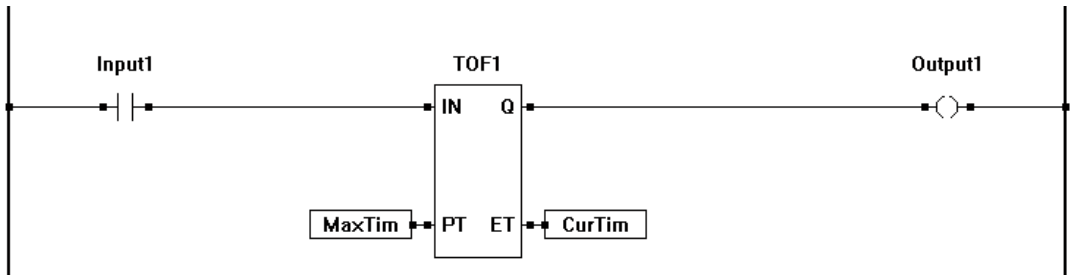
Outputs

- Q Boolean Output - True when IN is true and  $ET < PT$ .
- ET Timer Output - Current Elapsed timer value

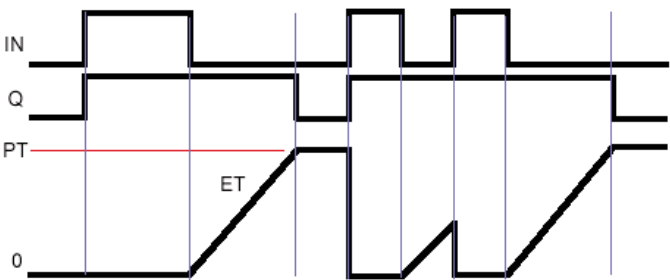
Description

The TOF (off delay timer / time delay on drop-out) is a programmable timer with a variable turn-off time. When the input (IN) input is true, the output (Q) is true. When the input (IN) sees a transition from true to false, the timer begins timing. When the elapsed time (ET) is equal to the preset time (PT), the output (Q) de-energizes (goes false). When the input (IN) sees a false to true to false transition, the timer is reset and begins timing again.

Example Circuit



Timing Diagram



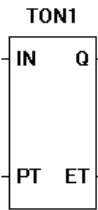
Related Functions: TON, TP

TON

Inputs

- IN Boolean Input - Timer sense/trigger input.
- PT Timer Input - Maximum time value (preset timer value).

Symbol



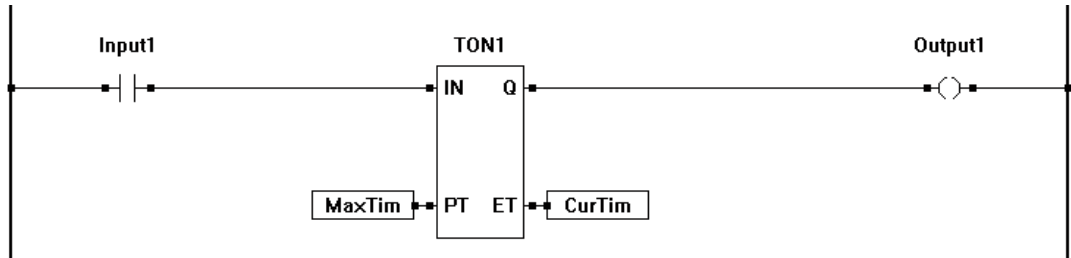
Outputs

- Q Boolean Output - True when IN is true and ET = PT.
- ET Timer Output - Current Elapsed timer value

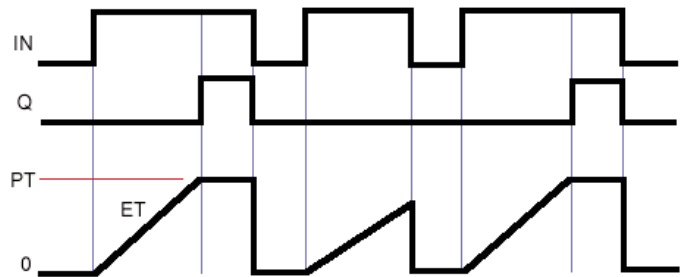
Description

The TON (on delay timer / time delay on pick-up) is a programmable timer with a variable turn-on time. When the input (IN) input is true, the timer begins timing. When the elapsed time (ET) is equal to the preset time (PT), the output (Q) energizes (goes true). When the input (IN) sees a true to false transition, the timer is reset and the output (Q) is de-energized (goes false).

Example Circuit



Timing Diagram



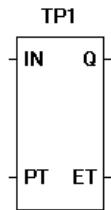
Related Functions: TOF, TP

TP

Inputs

- IN Boolean Input - Timer sense/trigger input.
- PT Timer Input - Maximum time value (preset timer value).

Symbol



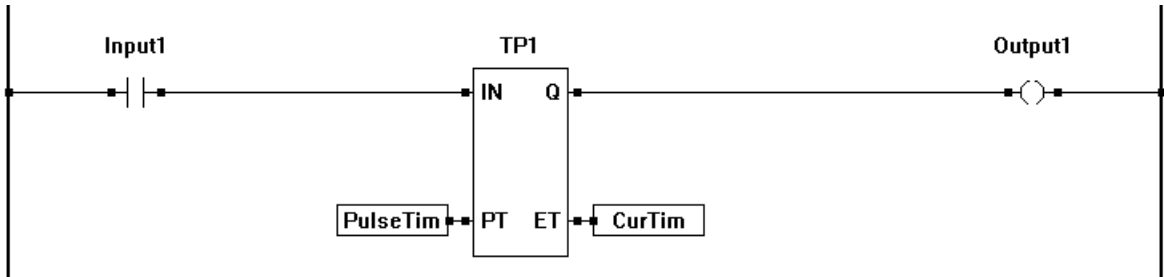
Outputs

- Q Boolean Output - True when IN is true and  $ET < PT$ .
- ET Timer Output - Current Elapsed timer value

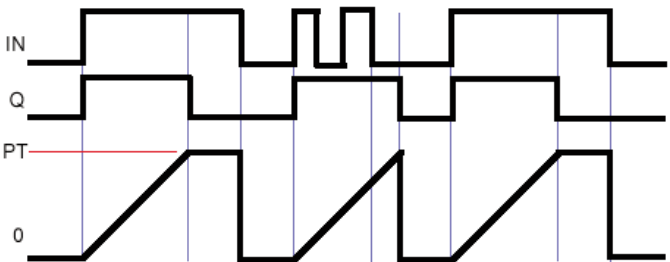
Description

The TP (pulse timer) is a programmable one-shot timer with a variable turn-on time. When the input (IN) input is true, the timer begins timing and the output (Q) is energized. When the elapsed time (ET) is equal to the preset time (PT), the output (Q) de-energizes (goes false). When the input (IN) goes from true to false, the timer is only reset if the elapsed time (ET) is equal to the preset time (PT). If they are not equal, the reset will not occur until they are equal (and IN must still be false).

Example Circuit



Timing Diagram



Related Functions: TON, TOF

UNLATCH (coil)

Symbol

Inputs

None



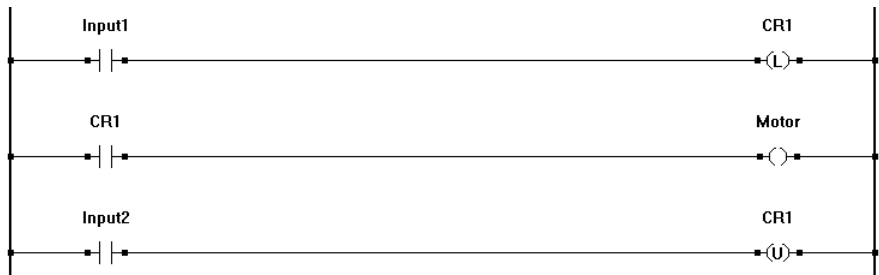
Outputs

None

Description

The UNLATCH coil acts like a direct coil, except when the coil is energized, it will cause the LATCHed coil to unlatch and de-energize.

Example Circuit



**Related Functions:** LATCH, DIRECT COIL, INVERTED COIL

XOR

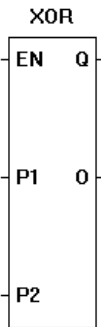
Inputs

- EN Boolean Function Enable Input - function is disabled if EN is false.
- P1 Integer Input - Output is bitwise XOR of inputs.
- P2 Integer Input - Output is bitwise XOR of inputs.

Outputs

- Q Boolean Function Enable Output - true when function is enabled.
- O Integer Output - Bitwise XOR of P1, P2 Inputs.

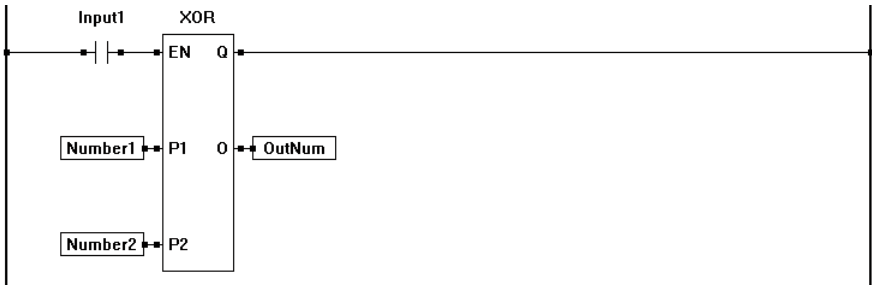
Symbol



Description

The XOR functions provides a bitwise exclusive OR function of the P1 and P2 inputs. The enable (EN) must be true for the XOR function to be enabled. The Q output is true when the XOR function is enabled.

Example Circuit



Related Functions: OR, AND

---

## DIVELBISS SOFTWARE LICENSE AGREEMENT

This Software License Agreement (the "Agreement") sets forth the terms by which Divelbiss Corporation, an Ohio corporation having a principal place of business at 9778 Mt. Gilead Road, Fredericktown, Ohio ("Divelbiss"), authorizes its bona fide licensees who have paid all applicable fees and accepted the terms of this Agreement (each a "Licensee") to use the Licensed Software (as defined below) provided herewith. Installing, using or attempting to install or use such Licensed Software or otherwise expressing assent to the terms herein constitutes acceptance of this Agreement. Any installation, use or attempted installation or use of such Licensed Software by any party other than a Licensee or otherwise in violation of this Agreement is expressly prohibited.

### Introduction

Whereas Divelbiss has developed certain modules of computer software known as "PLC ON A CHIP Kernel" and "EZ LADDER Toolkit"; and Licensee wishes to secure certain rights to use such software ; and Divelbiss is prepared to license such rights, subject to the terms and conditions of this Agreement; therefore, in consideration of the mutual covenants contained herein and intending to be legally bound hereby, Divelbiss and Licensee agree as follows:

### 1. Licensed Software

The PLC ON A CHIP Kernel and EZ LADDER Toolkit software, whether in source code or object code format, and all related documentation and revisions, updates and modifications thereto (collectively, "Licensed Software"), is licensed by Divelbiss to Licensee strictly subject to the terms of this Agreement.

### 2. License Grant

Divelbiss hereby grants to Licensee a nonexclusive, non-transferable license to use the Licensed Software as follows.

- (a) Except as otherwise provided herein, one (1) user may install and use on one (1) desktop personal computer and on one (1) portable personal computer the EZ LADDER Toolkit (i) to develop, test, install, configure and distribute certain applications on certain hardware devices such as programmable logic controllers (each a "Resulting Product"), and (ii) to configure the PLC ON A CHIP Kernel on designated processors, which shall constitute Resulting Products.
- (b) Licensee may copy the EZ LADDER Toolkit only for backup purposes.
- (c) Licensee may not amend, modify, decompile, reverse engineer, copy (except as expressly authorized in Section 2 of this Agreement), install on a network, or permit use by more than a single user, in whole or in part, the Licensed Software, or sublicense, convey or purport to convey any such right to any third party.
- (d) Licensee, Licensee's customers and others who obtain Resulting Products are expressly prohibited from using, in whole or in part, the Licensed Software and any Resulting Product, in any use or application (i) intended to sustain or support life; (ii) for surgical implant; (iii) related to the operation of nuclear facilities; (iv) in which malfunction or failure could result in death or personal injury; or (v) in environments otherwise intended to be fault-tolerant.

### 3. License Fee

- (a) Except when Licensee obtains the EZ LADDER Toolkit from an approved distributor or OEM pursuant to other fee arrangements, Licensee will pay to Divelbiss the license fee for the EZ LADDER Toolkit specified in the applicable Divelbiss price list, which is due and payable upon delivery of same.
- (b) If Licensee fails to make any payment when due, Divelbiss may, at its sole option, terminate Licensee's rights under this Agreement to use the Licensed Software. If Licensee fails to pay any balance within thirty (30) days after being notified by Divelbiss that payment is overdue, Divelbiss may take whatever steps it deems necessary to collect the balance, including referring the matter to an agency and/or suing for collection. All expenses and fees associated with the collection of an overdue balance, including costs and fees of collection and attorney's fees, shall be paid by Licensee. Overdue balances are subject to a monthly finance charge equal to the greater of [1.5]% or the maximum interest rate permitted by law times the unpaid balance.

### 4. Reporting

- (a) Upon request of Divelbiss, Licensee will provide a written report each quarter showing the number of Resulting Products produced, distributed or sold by Licensee during the previous calendar quarter, the parties (identified by name, address, etc.) to which they were distributed or sold, and the revenue received therefor.
  - (b) Divelbiss shall be entitled to commission or to conduct an audit of Licensee's books and records twice per year in order to verify the accuracy of reports regarding resulting Products made by Licensee to Divelbiss. Such audit shall be conducted during regular business hours at Licensee's facilities, and Licensee shall cooperate fully with in connection with such audit, making all facilities, records and personnel available upon request by Divelbiss or its representative.
-

---

## 5. Divelbiss Warranties

- (a) Divelbiss represents and warrants that (i) it is the owner of the Licensed Software, and (ii) this Agreement violates no previous agreement between Divelbiss and any third party.
- (b) Divelbiss further warrants that for a period of 90 days from the date this Agreement is accepted by Licensee, the EZ LADDER Toolkit will perform substantially in accordance with the accompanying documentation provided by Divelbiss, provided that the EZ LADDER Toolkit (i) has not been modified, (ii) has been maintained according to all applicable maintenance recommendations, (iii) has not been used with hardware or software or installed or operated in a manner inconsistent with any manuals or relevant system requirements provided by Divelbiss, and (iv) has not been subjected to abuse, negligence or other improper treatment, including, without limitation, use outside the operating environment or range of applications prescribed in any manuals or relevant system requirements provided by Divelbiss by Divelbiss. Provided that Licensee gives prompt written notice to Divelbiss of any alleged breach of the foregoing warranty and that such alleged breach can be reproduced by Divelbiss, Divelbiss will use commercially reasonable efforts to repair or replace the EZ LADDER Toolkit so that it performs as warranted, or, at its sole option, refund to Licensee a prorated share of the license fee paid by Licensee for the portion of the EZ LADDER Toolkit which caused the alleged breach of warranty. Licensee acknowledges that the foregoing represents Divelbiss's sole obligation and Licensee's sole remedy for any alleged breach of warranty regarding the EZ LADDER Toolkit.
- (c) Divelbiss expressly disclaims any and all warranties concerning any Resulting Products and any applications developed, tested, installed or distributed by Licensee using the Licensed Software, and Licensee expressly acknowledges that it is solely responsible for any and all Resulting Products and applications developed, tested, installed or distributed using the Licensed Software, and for any and all claims, damages, settlements, expenses and attorney's fees arising from the distribution or use of the PLC ON A CHIP Kernel or Resulting Products by Licensee, Licensee's customers or others.
- (d) DIVEBISS MAKES NO OTHER WARRANTIES OF ANY KIND WITH RESPECT TO THE LICENSED SOFTWARE OR THIS AGREEMENT, AND EXPRESSLY DISCLAIMS ALL OTHER WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NON-INFRINGEMENT.

## 6. Licensee Warranties

Licensee represents, warrants and covenants that:

- (a) Licensee has all necessary authority to enter into and to fulfill its obligations under this Agreement;
- (b) Licensee will comply with all federal, state and local laws and regulations applicable to the use or disposition of the Licensed Software, including without limitation all export laws and regulations;
- (c) Licensee shall be solely liable for all Resulting Products, any and all warranties on Resulting Products shall be made only by and on behalf of Licensee, and Licensee shall make NO representations or warranties on behalf of Divelbiss.
- (d) For the term of this Agreement and any renewal thereof, and for one (1) year thereafter, Licensee will not solicit or hire any of Divelbiss's employees.

## 7. Limitation of Liability

LICENSEE ACKNOWLEDGES AND AGREES THAT NEITHER DIVEBISS NOR ITS SUPPLIERS, EMPLOYEES OR AFFILIATES WILL BE LIABLE FOR ANY LOSS OF PROFITS, LOSS OF BUSINESS OR GOODWILL, LOSS OF DATA OR USE OF DATA, INTERRUPTION OF BUSINESS, NOR FOR ANY INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES OF ANY KIND UNDER, ARISING OUT OF, OR RELATED TO THE SUBJECT MATTER OF THIS AGREEMENT (SPECIFICALLY INCLUDING ANY LOSS TO OR DAMAGES OF LICENSEE'S CUSTOMERS, OF ANY SORT WHATSOEVER), HOWEVER CAUSED, WHETHER ANY SUCH CLAIM SOUNDS IN CONTRACT, TORT, STRICT LIABILITY OR OTHER LEGAL OR EQUITABLE THEORY, EVEN IF DIVEBISS HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH LOSS. IN NO EVENT WILL DIVEBISS'S LIABILITY UNDER, ARISING OUT OF OR RELATED TO THE SUBJECT MATTER OF THIS AGREEMENT EXCEED THE AMOUNT RECEIVED BY DIVEBISS FROM LICENSEE UNDER THIS AGREEMENT DURING THE NINETY (90) DAY PERIOD PRECEDING THE EVENT GIVING RISE TO SUCH LIABILITY, OR THE AMOUNT OF A SINGLE-USER LICENSE FEE FOR THE EZ LADDER TOOLKIT, WHICHEVER IS GREATER.

## 8. Indemnification

- (a) Subject to the limitations of Section 7 of this Agreement, Divelbiss will indemnify Licensee from and against liability for any judgment finally awarded by a court of competent jurisdiction against Licensee based upon a claim that the EZ LADDER Toolkit infringes any current U.S. patent or copyright of a third party, provided that Divelbiss is promptly notified of any such threats, claims or proceedings, afforded the opportunity to intervene in any such proceeding and given sole control over the defense of such claim, including all negotiations of any prospective settlement or compromise, and that Licensee gives all cooperation and assistance requested by Divelbiss in connection with same; and provided further that the foregoing obligation of Divelbiss does not apply with respect to any Resulting Products or any hardware, software (including the Licensed Software) or components thereof (i) not supplied by Divelbiss, (ii) made or modified in whole or in part by Licensee or according to Licensee's specifications, (iii) otherwise modified after delivery, (iv) combined with other hardware, software,
-

---

products or processes by Licensee (including in creating Resulting Products) where such claim could have been avoided absent such combination, (v) insofar as Licensee continues allegedly infringing activity after being notified thereof or informed of steps or modifications that would have avoided the alleged infringement, or (vi) used by Licensee in violation of the terms of this Agreement.

- (b) Licensee will defend, indemnify and hold Divelbiss harmless from and against any and all losses, liabilities, judgments, damages and claims against Divelbiss obtained or asserted by any third party (including any allegation of infringement or violation of proprietary rights), and all related costs, including attorney fees, incurred by Divelbiss, arising or resulting from or related to i) Licensee's use, modification or adaptation of the Licensed Software, including to create any application or any Resulting Product, ii) the operation or performance, or Licensee's or any third party's use, of any Resulting Product, iii) any breach by Licensee of any representation or warranty made by Licensee related to the Licensed Software or any Resulting Product, or iv) any breach by Licensee of any of its obligations under this Agreement.
- (c) In the event that any claim of infringement under Section 8(a) above is, or in Divelbiss's sole judgment is likely to be, substantiated, Divelbiss may, at its sole discretion, use commercially reasonable efforts to i) obtain a license from the third party for Licensee to continue using the allegedly infringing feature or aspect of the EZ LADDER Toolkit; ii) replace or modify the allegedly infringing feature or aspect of the EZ LADDER Toolkit to avoid such infringement; or iii) terminate this Agreement and the license hereunder and refund a prorated portion of the initial license fee paid by Licensee for the allegedly infringing feature or aspect of the EZ LADDER Toolkit .

## **9. Modification of Licensed Software**

- (a) Divelbiss may, from time to time, at its sole discretion and without further notice to Licensee, make, and at its further discretion distribute to Licensee, modifications to the Licensed Software. In the event that Licensee fails to install such a modification when so advised by Divelbiss, Divelbiss shall be relieved of any obligation pursuant to the limited warranty set forth in Section 5 hereof. Should Licensee request modifications to the Licensed Software, Divelbiss may charge for and make such changes subject to the terms of a separate agreement between the parties.
- (b) Licensee may not modify the Licensed Software or engage any third party to modify the Licensed Software without the express, written consent of Divelbiss. Any and all modifications made to the Licensed Software, whether by Licensee or any third party, and all rights therein are hereby assigned to and shall be the sole and exclusive property of Divelbiss.

## **10. Ownership of Licensed Software**

- (a) Licensee acknowledges that, subject only to the license specifically granted herein, all right, title, and interest in and to the Licensed Software, all revisions and copies thereof provided to or created by Licensee and all modifications thereof, by whomever made, are and shall remain the sole and exclusive property of Divelbiss.
- (b) LICENSEE ACKNOWLEDGES THAT VARIOUS ASPECTS AND FEATURES OF THE LICENSED SOFTWARE MAY BE PROTECTED UNDER APPLICABLE PATENT, COPYRIGHT, TRADEMARK AND TRADE SECRET LAW AND THAT, EXCEPT AS EXPRESSLY AUTHORIZED IN WRITING BY DIVELBISS, LICENSEE MAY NOT USE, DISCLOSE OR REPRODUCE OR DISTRIBUTE ANY COPIES OF THE LICENSED SOFTWARE, IN WHOLE OR IN PART, NOR AUTHORIZE OR PERMIT OTHERS TO DO SO.
- (c) Licensee further acknowledges that any applications made by Licensee using the Licensed Software, including any incorporated into Resulting Products, are derivative works made solely with the authorization of Divelbiss, in consideration for which Licensee agrees to provide, upon request from Divelbiss, copies of all such applications to Divelbiss and grants to Divelbiss a perpetual, irrevocable, royalty-free license to copy and use such applications so long as Divelbiss is not competing with Licensee.
- (d) Licensee shall not, nor will it assist others in attempting to, decompile, reverse engineer or otherwise re-create the source code for or functionality of the Licensed Software. Licensee shall not use the Licensed Software for the purpose of developing any similar or competing product, or assisting a third party to develop a similar or competing product.
- (e) At no expense to Divelbiss, Licensee will take any action, including executing any document, requested by Divelbiss in order to secure, perfect or protect the rights of Divelbiss in the Licensed Software or Confidential Information (as hereinafter defined).

## **11. Confidentiality**

Except as expressly provided in this Agreement, Licensee shall not disclose or permit disclosure to any third parties the Licensed Software (including object code, source code and documentation) or any other confidential information provided by Divelbiss ("Confidential Information"). Further, Licensee will use all reasonable precautions and take all steps necessary to prevent any Confidential Information from being acquired, in whole or in part, by any unauthorized party, will use Confidential Information solely in furtherance of this Agreement, and will permit access to any Confidential Information only by those employees of Licensee with a legitimate "need to know." In the event that Licensee learns or has reason to believe that Confidential Information has been disclosed or is at risk of being disclosed to any unauthorized party, Licensee will immediately notify Divelbiss thereof and will cooperate fully with Divelbiss in seeking to protect Divelbiss's rights in the Confidential Information.

---



---

## **12. Term and Termination**

- (a) This Agreement shall remain in effect from the date it is accepted until terminated as provided below.
- (b) Divelbiss may terminate this Agreement and all license rights hereunder upon the occurrence of any of the following:
  - (i) Licensee fails to cure any material breach of this Agreement within thirty (30) days after receiving notice of such breach;
  - (ii) Licensee becomes insolvent or unable to pay its debts, makes an assignment for the benefit of creditors, ceases to be a going concern, files for protection of the bankruptcy laws, becomes the subject of any involuntary proceeding under federal bankruptcy laws or has a receiver or trustee appointed to manage its assets;
  - (iii) Licensee consolidates or merges into or with any other entity or entities or sells or transfers all or substantially all of its assets; or
  - (iv) Following ninety (90) days written notice of termination to Licensee.
- (c) Licensee may terminate this Agreement and all licenses hereunder in the event that Divelbiss fails to cure any material breach of this Agreement within thirty (30) days after receiving notice of such breach.
- (d) Any fees or expenses payable by Licensee to Divelbiss shall not be reduced or otherwise affected by termination of this Agreement. In the event of termination of this Agreement for any reason, neither party shall be liable to the other on account of loss of prospective profits or anticipated sales, or on account of expenditures, inventories, investments, or commitments.
- (e) Upon termination of this Agreement for any reason, Licensee will immediately return to Divelbiss or, upon instruction from Divelbiss, destroy all copies of the Licensed Software (including all code, documentation, manuals, etc.) and all Confidential Information in its possession, and will certify in writing to Divelbiss that it has done so.
- (f) All provisions regarding ownership, confidentiality, proprietary rights, payment of fees and royalties, indemnification, disclaimers of warranty and limitations of liability will survive termination of this Agreement.

## **13. Assignment and Sublicensing**

This Agreement, the license granted hereunder and the Licensed Software may not be assigned, sublicensed or otherwise transferred or conveyed by Licensee to any third party without the express, written consent of Divelbiss.

## **14. Dispute Resolution**

- (a) In the event of any dispute arising between the parties related to the subject matter of this Agreement, except regarding the payment of fees under Sections 3 or 15 of this Agreement or as provided in Subsection (b) below ("Dispute"), the parties agree to attempt to resolve such Dispute according to the procedures set forth below.
  - (i) In the event either Divelbiss or Licensee notifies the other party of a Dispute, representatives of each party with adequate authority to settle such Dispute will promptly engage in direct negotiations. If such representatives are unable to resolve such Dispute within ten (10) business days after commencing negotiations, or twenty (20) business days after the initial notice of Dispute, then either party may initiate mediation of the Dispute as provided in Subsection (a)(ii) below.
  - (ii) In the event either party initiates mediation of the Dispute (by sending a written notice of mediation to the other party), then the Dispute shall be subject to mediation in Mt. Vernon, Ohio before a single mediator (to be proposed, in the first instance, by the party initiating mediation) who will be reasonably familiar with the computer industry and mutually acceptable to the parties. The parties agree to participate in such mediation in good faith, through representatives with due authority to settle any such Dispute. If such representatives are unable to resolve such Dispute within twenty (20) business days after commencing mediation, then each party may pursue whatever further recourse it deems necessary to protect its rights under this Agreement.
- (b) Licensee agrees that any violation of this Agreement related to the Licensed Software or Confidential Information, specifically including Divelbiss's proprietary rights therein, is likely to result in irreparable injury to Divelbiss. Accordingly, notwithstanding any other provision of this Agreement to the contrary, Licensee agrees that Divelbiss shall be entitled to all appropriate relief from any court of competent jurisdiction, whether in the form of injunctive relief and/or monetary damages, to protect its proprietary rights in the Licensed Software and Confidential Information.

## **15. Maintenance and Support**

- (a) In consideration of the payment of annual maintenance and support fees by or on behalf of Licensee (payable for the first year with the license fee, and thereafter annually at least thirty (30) days before the anniversary date of this Agreement),
-

---

Divelbiss will provide maintenance and support of the EZ LADDER Toolkit, in the form of (i) such periodic corrections, updates and revisions to the EZ LADDER Toolkit as Divelbiss, in its sole discretion, may from time to time elect to release, and (ii) responses to inquiries submitted by Licensee by email to Divelbiss at sales@divelbiss.com.

(b) The maintenance and support fee is specified in the applicable Divelbiss price list.

## **6. General**

- (a) This agreement constitutes the entire agreement between the parties relating to the Licensed Software and the subject matter hereof, supersedes all other proposals, quotes, understandings or agreements, whether written or oral, and cannot be modified except by a writing signed by both Licensee and Divelbiss.
- (b) In the event of any conflict between the terms of this Agreement and any purchase order or similar documentation prepared by Licensee in connection with the transactions contemplated herein, this Agreement shall govern and take precedence, notwithstanding Divelbiss's failure to object to any conflicting provisions.
- (c) Notwithstanding anything to the contrary herein, except for payment obligations under Sections 3 or 15, neither party shall be liable for any failure of performance beyond its reasonable control.
- (d) Except as otherwise provided, this Agreement will be subject to and construed in accordance with the laws of the State of Ohio (U.S.A.) without regard to its conflict of laws provisions. Exclusive venue for any legal action between the Parties arising out of or related to this Agreement or the subject matter hereof will be in the state or federal courts located or having jurisdiction in Knox County, Ohio (U.S.A.), which the Parties expressly acknowledge to have personal jurisdiction over them. The 1980 UN Convention on the International Sale of Goods (CISG) will not apply hereto.
- (e) No waiver by either party of a breach of this Agreement shall operate or be construed as a waiver of any subsequent breach.
- (f) The invalidity, illegality or unenforceability of any provision of this Agreement shall not affect the remainder of the Agreement, and this Agreement shall be construed and reformed without such provision, provided that the ability of neither party to obtain substantially the bargained for performance of the other shall have thereby been impaired.
- (g) All notices, consents and other communications between the parties shall be in writing and shall be sent by (i) first class mail, certified or registered, return receipt requested, postage prepaid, (ii) electronic facsimile transmission, (iii) overnight courier service, (iv) telegram or telex or (v) messenger, to the respective addresses that the parties may provide.
- (h) Licensee shall be deemed an independent contractor hereunder, and as such, shall not be deemed, nor hold itself out to be, an agent or employee of Divelbiss. Under no circumstances shall any of the employees of a party hereto be deemed to be employees of the other party for any purpose. This Agreement shall not be construed as authority for either party to act for the other party in any agency or other capacity, or to make commitments of any kind for the account of or on behalf of the other except to the extent and for the purposes provided herein.
- (i) LICENSEE ACKNOWLEDGES THAT IT HAS READ THIS AGREEMENT, UNDERSTANDS IT, AND AGREES TO BE BOUND BY ITS TERMS AND CONDITIONS.